GenSpect process algebra

van Weerdenburg, M.J.

Award date:
2004

TECHNISCHE UNIVERSITEIT EINDHOVEN
Department of Mathematics and Computing Science

MASTER'S THESIS

GenSpect Process Algebra

by

Muck van Weerdenburg

Supervisor: prof. dr. ir. J.F. Groote

Eindhoven, July 2004

GenSpect Process Algebra

Muck van Weerdenburg
M.J.v.Weerdenburg@student.tue.nl

Abstract

A process algebra (GenSpect) is introduced as a basis for a set of languages used for modelling and verification. The given axiomatisation for this algebra is proven to be sound and (relatively) complete. Some examples are given to demonstrate the application of this algebra, as well as a number of alphabet axioms for more efficient linearisation.

1 Introduction

To model and verify properties of systems there are numerous languages, but often (if not always) such languages allow only certain systems and properties, and are not always easy to use. We believe it would be far more efficient if one could model a system in a framework and have the possibility to easily verify whatever property the system should have. To do so, a good approach would probably be choosing a common set of languages, which allows to model systems in such a way that all specific qualities of the languages can be used efficiently. (We consider it not very likely to find the solution in a single language, as it has to incorporate every possible aspect of a system and still be easy/efficient to use.)

Such a set most likely has some high level component language, defining the components of a system and their interaction. These components (and possibly their interactions as well) are then specified in for example a Petri-net[7] or some algebraic equations. The languages that are (in some sense) higher level than another should ideally be translatable to the lower level language(s), adding verification possibilities of these languages. The other way around would be nice as well, but might not always be possible.

The process algebra presented in this document is an effort to provide a basic (lower) level language of a set of languages (including for example Petri-nets and Guarded Command Language[9]) as described above. The essence of it lies in the use of multiactions, which is, for example, necessary to (nicely) model the behaviour of processors in (coloured) Petri-nets.

We will now informally describe the components of our algebra. A multiaction is a bag of actions (possibly with data) that execute together. We write a multiaction of actions a, $b(d)$ and c as $\langle a, b(d), c\rangle$ (or $\langle b(d), a, c\rangle$ as order has no meaning in bags). Often we write multiactions that consist of only one action as that action alone (i.e. $a(d)$ instead of $\langle a(d)\rangle$). We can combine such multiactions with the common operators $\cdot$ and $+$ to form a sequence of multiactions or a nondeterministic choice between multiactions respectively. The alternative quantification $\sum$ allows processes described by the (possibly infinite) alternative composition $\langle a(d_0)\rangle + \langle a(d_1)\rangle + \ldots$ to be written as $\sum_{d:D}\langle a(d)\rangle$, where D is a set $\{d_0, d_1, \ldots\}$. To select only certain alternatives there is a conditional operator $\rightarrow$ which deadlocks unless a condition on data variables is true (e.g. $\sum_{n:\mathbb{N}} n < 25 \rightarrow \langle a(n)\rangle$ which can (only) choose to execute $\langle a(n)\rangle$ for all natural numbers n smaller than 25). To denote inaction or deadlock we write δ.

For parallel composition we have the merge $\|$ which interleaves and/or synchronises multiactions. For example, $(a+b) \parallel (c\cdot d)$ has the same behaviour as $(a+b)\cdot c\cdot d + c\cdot((a+b)\cdot d + d\cdot(a+b) + (\langle a, d\rangle +$

$\langle b,d\rangle))+(\langle a,c\rangle+\langle b,c\rangle)\cdot d$. The communication operator Γ allows explicit specification of (two or more) actions that communicate with each other (e.g. $a|b \to c$, which means that a and b communicate to c), besides just being synchronised, which is only possible if two actions have equivalent data parameters. So $\Gamma_{\{a|c\to d\}}(\langle a(n),b,c(n)\rangle+\langle a(m),c(o)\rangle)$ is the same as $\langle b,d(n)\rangle+\langle a(m),c(o)\rangle$ (with $m\neq o$). To limit the behaviour of a process we have a restriction operator ∇ that specifies precisely which multiactions are allowed with a set of action sequences (e.g. a or $b|c|c$). If one wishes that in the parallel composition of a,b and c action a does not execute synchronised with another action and b and c must synchronise, one can write $\nabla_{\{a,b|c\}}(a \parallel b \parallel c)$, which behaves as $a\cdot\langle b,c\rangle+\langle b,c\rangle\cdot a$.

The filter operator ∂_H prohibits actions in its set parameter H from executing (e.g. $\partial_{\{a\}}(a+b\cdot\langle a,c\rangle)$, which behaves as $b\cdot\delta$), the hiding operator τ_I makes actions in I invisible (e.g. $\tau_{\{a\}}(\langle a,b\rangle)$ becomes $\langle b\rangle$) and the renaming operator ρ renames actions (e.g. $\rho_{\{a\to b\}}(a)$ becomes b). The special case of the empty multiaction $\langle\rangle$ is called a silent step, which we often write as τ. Finally we have process variables with which we can write equations as $X=a\cdot X$ to denote the process that can do infinitely many a's.

Note that the data expressions used with $\to$ and as parameters require an additional data algebra. The specifics of this data algebra, besides those described in the next section, are not relevant in specifying the process algebra.

In the rest of this document we first describe what we need to know of the data algebra, after which we give the formal syntax and semantics of our process algebra. Following this we give an axiomatisation for the given semantics and prove it sound and complete. Finally we give some examples of the use of the algebra and alphabet axioms to make automated calculation more efficient.

2 Data model

The data expressions we use in our process expressions are conforming some data model. Although we need not know all details of it, we need to define what we use of it. In specific, we need to know when a data expression is true or false to be able to define the conditional operator $\to$ and we use boolean expressions in our proof of the completeness of our axiomatisation. The following definitions specify as much as we need to know about the data algebra.

Definition 2.1. Let $\mathbb{D}$ be a set of *types* and let $\mathbb{F}$ be a set of *function symbols*. Also, let $\mathbb{S}$ be *function specifications* (i.e. elements of $\mathbb{S}$ have the form $\langle F,D_1,\ldots,D_n,D\rangle$, with $F\in\mathbb{F}$ and $D_1,\ldots,D_n,D\in\mathbb{D}$) and $\mathbb{V}=\bigcup_{D\in\mathbb{D}}V_D$, with sets V_D of variables of type D. We call $\langle\mathbb{D},\mathbb{F},\mathbb{S},\mathbb{V}\rangle$ a *data signature.*

Definition 2.2. Let $\Sigma=\langle\mathbb{D},\mathbb{F},\mathbb{S},\mathbb{V}\rangle$ be a data signature. The set T_Σ of *data terms* over Σ are defined as follows (see Section 3 for notation):

$$T_\Sigma ::= \mathbb{F} \mid \mathbb{F}(T_\Sigma\ (,T_\Sigma)^*) \mid \mathbb{V}$$

Definition 2.3. Let $\Sigma=\langle\mathbb{D},\mathbb{F},\mathbb{S},\mathbb{V}\rangle$ be a data signature. Let $\mathcal{D}$ be a set of sets, corresponding to the types in $\mathbb{D}$ (i.e. for every $D\in\mathbb{D}$ there is a set $D_\mathcal{M}\in\mathcal{D}$ with the elements of type D). Also, let $\mathcal{F}$ be a set of functions on elements of $\mathcal{D}$ (i.e. for every function specification $\langle F,D_1,\ldots,D_n,D\rangle\in\mathbb{F}$ there is exactly one $F_\mathcal{M}\in\mathcal{F}$ with $D_{\mathcal{M}_1},\ldots,D_{\mathcal{M}_n},D_\mathcal{M}\in\mathcal{D}$ and $F_\mathcal{M}: D_{\mathcal{M}_1}\times\ldots\times D_{\mathcal{M}_n}\to D_\mathcal{M}$). We call $\langle\mathcal{D},\mathcal{F}\rangle$ a data *model* for Σ.

Definition 2.4. Let Σ be a data signature and let $\mathcal{M}$ be a data model for Σ. We call a pair $\langle\Sigma,\mathcal{M}\rangle$ a *data algebra.*

Definition 2.5. Let $\mathcal{A}=\langle\langle\mathbb{D},\mathbb{F},\mathbb{S},\mathbb{V}\rangle,\langle\mathcal{D},\mathcal{F}\rangle\rangle$ be a data algebra and $v:\mathbb{V}\to\bigcup_{D\in\mathbb{D}}D_\mathcal{M}$. We call v a *valuation* of $\mathcal{A}$, which maps variables to (corresponding) values.

Definition 2.6. Let $\mathcal{A} = \langle\langle \mathbb{D}, \mathbb{F}, \mathbb{S}, \mathbb{V}\rangle, \langle \mathcal{D}, \mathcal{F}\rangle\rangle$ be a data algebra, v a valuation of $\mathcal{A}$ and let $x \in \mathbb{V}$, $\langle F, D_1, \ldots, D_n, D\rangle \in \mathbb{S}$. Also, let data terms t_i be of type D_i (for $1 \leq i \leq n$) and $F_{\mathcal{M}} \in \mathcal{F}$ corresponding to F. We extend the valuation v to valuation $\overline{v}$ to data terms with the following definition:

$$\begin{array}{lcl} \overline{v}(x) & = & v(x) \\ \overline{v}(F) & = & F_{\mathcal{M}}() \\ \overline{v}(F(t_1, \ldots, t_n)) & = & F_{\mathcal{M}}(\overline{v}(t_1), \ldots, \overline{v}(t_n)) \end{array}$$

Definition 2.7. Let $\Sigma = \langle \mathbb{D}, \mathbb{F}, \mathbb{S}, \mathbb{V}\rangle$ be a data signature and $D \in \mathbb{D}$. We call a data term d (described by T_Σ) *well-typed* and of type D if, and only if, the following holds:

- if $d \in V_D$, or
- if $d = F$, with $\langle F, D\rangle \in \mathbb{S}$.
- if $d = F(t_1, \ldots, t_n)$, with t_i well-typed data terms of type D_i (for $1 \leq i \leq n$ and $D_i \in \mathbb{D}$) and $\langle F, D_1, \ldots, D_n, D\rangle \in \mathbb{S}$.

In the rest of this document we assume that all data terms are well-typed.

As we are going to use booleans (B), we assume the following on our data model: $B \in \mathbb{D}$, $B_{\mathcal{M}} = \{\, f\,, t\,\}$ (with $f \neq t$), $B_{\mathcal{M}} \in \mathcal{D}$ and $\neg : B_{\mathcal{M}} \to B_{\mathcal{M}}$, $\vee, \wedge, \Rightarrow, \Leftrightarrow$: $B_{\mathcal{M}} \times B_{\mathcal{M}} \to B_{\mathcal{M}}$, all in $\mathcal{F}$. These functions (and constants) correspond to the following function symbols: $\boldsymbol{f}$, $\boldsymbol{t}$, $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$. These functions are defined by $\neg\, f = t$, $\neg\, t = f$ and (with the first parameter vertically):

$\vee$	f	t
f	f	t
t	t	t

$\wedge$	f	t
f	f	f
t	f	t

$\Rightarrow$	f	t
f	t	t
t	f	t

$\Leftrightarrow$	f	t
f	t	f
t	f	t

Definition 2.8. Let $\mathcal{A}$ be a data algebra with signature Σ and valuation v. Let E be a term of type B in our data model. We say that a data term E over Σ of type B is true if, and only if, for all valuations v the following holds: $\overline{v}(E) = t$. We write this as $\mathcal{A} \models E$.

For the rest of this document we assume there is some data algebra $\mathcal{A} = \langle \Sigma, \mathcal{M}\rangle$ (containing booleans as described above) with valuation v and axiomatisation with relation $\doteq_d$ (with $\overline{v}(t) = \overline{v}(u)$ if $t \doteq_d u$, for all valuations v).

3 Syntax

For the description of the syntax of GenSpect we use a BNF like notation. We use | to separate alternatives, [ ] to specify an optional part and ( )* for zero or more occurrences of an expression. As terminals we use sets, meaning that any element of that set is allowed at that position.

Definition 3.1. Let $\mathcal{N}_A$ the set of action names, X the set of process variables, $\mathbb{V}$ the set of data variables and $\mathbb{D}$ the set of data types. Also, let T_B be a boolean term. Our process algebra GenSpect (G) has the following syntax. (T describes variable typings, L_D lists of data terms, A_M (multi)actions, N "communication action names", V sets of N (for ∇), C sets of communication relations (for Γ), IH sets of action names (to be hidden with τ or filtered with ∂), R renaming functions (for ρ), T_P GenSpect terms and E GenSpect expressions.)

$$\begin{array}{lll}
T & ::= & \mathbb{V} : \mathbb{D} \\
L_D & ::= & T_D \;(,T_D)^* \\
A & ::= & \mathcal{N}_A \mid \mathcal{N}_A(L_D) \\
A_M & ::= & A \mid \langle [A\;(,A)^*] \rangle \\
N & ::= & \mathcal{N}_A \;(|\mathcal{N}_A)^* \\
N^+ & ::= & \mathcal{N}_A | \mathcal{N}_A \;(|\mathcal{N}_A)^* \\
V & ::= & \{N\;(,N)^*\} \\
C & ::= & \{N^+ \rightarrow \mathcal{N}_A | \tau \;(,N^+ \rightarrow \mathcal{N}_A | \tau)^*\} \\
IH & ::= & \{\mathcal{N}_A\;(,\mathcal{N}_A)^*\} \\
R & ::= & \{\mathcal{N}_A \rightarrow \mathcal{N}_A\;(,\mathcal{N}_A \rightarrow \mathcal{N}_A)^*\} \\
T_P & ::= & A_M \mid \delta \mid \tau \mid T_P + T_P \mid T_P \cdot T_P \mid T_B \rightarrow T_P \mid \sum_T T_P \mid T_P \parallel T_P \mid T_P \mathbin{\lfloor\!\lfloor} T_P \mid T_P | T_P \mid X \mid X([L_D]) \mid \\
 & & (T_P) \mid \nabla_V(T_P) \mid \Gamma_C(T_P) \mid \partial_{IH}(T_P) \mid \tau_{IH}(T_P) \mid \rho_R(T_P) \\
E & ::= & X = T_P \mid X([T\;(,T)^*]) = T_P \mid E, E
\end{array}$$

As not all (infix) operators are necessarily enclosed within parentheses (e.g. we may write $a \cdot b + c$), terms can be interpreted in different ways (e.g. $(a \cdot b) + c$ or $a \cdot (b + c)$). To overcome this problem we give all infix operators and $\sum$ a binding strength with the meaning that if, for example, $\cdot$ binds stronger than $+$ we interpret $a \cdot b + c$ as $(a \cdot b) + c$. The order of the operators (in decreasing strength) is as follows: $\cdot$, $\rightarrow$, $|$, $\lfloor\!\lfloor$, $\parallel$, $\sum$, $+$. We assume all infix operators are right associative.

Instead of writing the sequence of terms $t_1, t_2, \ldots, t_n$ (e.g. the data terms in an action or the actions in a multiaction) we often write $\overrightarrow{t}$. The set of all actions A is defined by $\mathcal{N}_A \cup \{a(t_1, t_2, \ldots, t_n) \mid a \in \mathcal{N}_A \wedge n \in \mathbb{N} \wedge 1 \leq i \leq n \wedge t_i \in T_\Sigma\}$, and the set of all multiactions $\mathbb{A}$ is defined by $\{\langle \overrightarrow{a} \rangle \mid \overrightarrow{a} \in \overrightarrow{A}\}$. Although we allow the action a in our syntax, in this document we only talk about that action as being $a()$. Similarly we consider expressions $X = t$ as being of the form $X() = t$. We also write α instead of the action $\langle \overrightarrow{a} \rangle$.

To be able to reason about terms of our language, we introduce some sets and notations. The set V_P, with elements $x, y, \ldots$, consists of process variables, for which we assume that in terms, that are substituted for such a process variable, no free variables (i.e. not bound by any operator in the term) may occur. For the set of GenSpect terms (described by) T_P we have elements $t, u, \ldots$ and process-closed terms $p, q, \ldots$ in T_{pc} (terms that do not have any process variables in them).

Note that this syntax allows one to write sets (R and C) that can contain elements with the same left hand side (e.g. $\{a \rightarrow b, a \rightarrow c\}$). This should not be possible as the meaning of these sets are meant to be functions. Therefore we put the restriction on this syntax that in the sets described by R and C no left hand side of an element may be the same as the left hand side of another.

Another restriction on C is that left hand sides must be disjoint (i.e. $\{a|b \rightarrow c, d|b \rightarrow e\}$ is not allowed as b occurs in both left hand sides). This is to ensure unicity of the communication (see the semantics later on).

4 Operational Semantics

Now we have a formal syntax, we give meaning to terms by the following semantics.

Definition 4.1. Let T and T' be some sets. We call $F \subseteq T \times T'$ a *function* from T to T' if, and only if, $\forall_{t \in T}(\forall_{t'_1, t'_2 \in T'}(\langle t, t'_1 \rangle, \langle t, t'_2 \rangle \in F \Rightarrow t'_1 = t'_2))$ holds. The subset of functions in the powerset of $T \times T'$ is written as $T \rightarrow T'$ (i.e. $T \rightarrow T' \subseteq \mathcal{P}(T \times T')$, with $F \in T \rightarrow T'$ if, and only if, F is a function from T to T').

Note that we often write a colon (:) instead of $\in$.

Definition 4.2. Let T and T' be sets and $F : T \rightarrow T'$. The *domain* of F, notation *dom*(F), is defined as follows:

$$dom(F) = \{t \mid \exists_{t' \in T'}(\langle t, t' \rangle \in F)\}$$

Definition 4.3. Let T and T' be sets and $F : T \rightarrow T'$. Also, let $t \in dom(F)$. We define the *function application*, notation $F(t)$ (i.e. F applied to t), as follows:

$$F(t) = t' \quad with\ \langle t, t' \rangle \in F$$

Definition 4.4. Let T and T' be sets with $T \subseteq T'$ and $F : T \rightarrow T'$. The *domain extension* F^+ of F is defined by:

$$F^+ = F \cup \{\langle t, t \rangle \mid t \in T \wedge t \notin dom(F)\}$$

We describe the semantics of GenSpect by looking at which actions can be executed by a process and what (process) the result of such an action is. This is expressed in one of the following definitions, but first we need to express that the order of actions within a multiaction is irrelevant (i.e. $\langle \vec{a}, \vec{b} \rangle$ and $\langle \vec{b}, \vec{a} \rangle$ are the same). We do this by interpreting them as bags of actions.

Definition 4.5. Let S be a set. A *bag* B of S is a function $S \rightarrow \mathbb{N}$, with the meaning that an element $s \in S$ occurs $B(s)$ times in the bag B. We write a bag as $[s_1, s_2, \ldots, s_n]$, such that an element $s \in S$ occurs $B(s)$ times exactly. We write $\mathbb{B}(S)$ to denote the set of all bags of S (i.e. $B \in \mathbb{B}(S)$).

Definition 4.6. Let S be a set and let B_1 and B_2 be bags of S (i.e. $B_1, B_2 \in \mathbb{B}(S)$). The *joining* operator $\oplus_S : \mathbb{B}(S) \times \mathbb{B}(S) \rightarrow \mathbb{B}(S)$ is defined by $(B_1 \oplus_S B_2)(s) = B_1(s) + B_2(s)$ for all $s \in S$.

Definition 4.7. Let S be some set and $B, C \in \mathbb{B}(S)$, $T \subseteq S$ and $s \in S$. We introduce the *intersection* $\cap : \mathbb{B}(S) \times \mathcal{P}(S) \rightarrow \mathcal{P}(S)$, the *bag inclusion* $\subseteq : \mathbb{B}(S) \times \mathbb{B}(S) \rightarrow B$, the *element test* $\in : S \times \mathbb{B}(S) \rightarrow B$ and the *size function* $|\,| : \mathbb{B}(S) \rightarrow \mathbb{N}$ as follows:

$$\begin{array}{lcll}
s \in B & = & B(s) > 0 & \\
[] \cap T & = & \emptyset & \\
([s] \oplus_S B) \cap T & = & B \cap T & if\ s \notin T \\
([s] \oplus_S B) \cap T & = & \{s\} \cup (B \cap T) & if\ s \in T \\
B \subseteq C & = & \forall_{s \in S}(B(s) \leq C(s)) & \\
|B| & = & \sum_{s \in S} B(s) &
\end{array}$$

Note that a bag $B \in \mathbb{B}(S)$ can be considered to be of the following structure: $B = []$ or $B = [s] \oplus_S C$, with $s \in S$ and $C \in \mathbb{B}(S)$. We usually write $\oplus$ instead of $\oplus_S$, if this cannot lead to any confusion. In this document we only consider finite bags.

Definition 4.8. We call $\boldsymbol{A} = \{a(\vec{d}) \mid a \in \mathcal{N}_A \wedge \vec{d} \in \overrightarrow{D_\mathcal{M}}\}$ the set of semantic (parameterised) actions (i.e. $a(\vec{d}) \in \boldsymbol{A}$).

Let t be some syntactic term and v a valuation, which maps variables to some corresponding semantic values/terms. The *interpretation* of t will be written as $[\![t]\!]^v$. The specific value of $[\![t]\!]^v$ will be separately defined for all (needed) forms of t. We mostly write $[\![t]\!]$ instead of $[\![t]\!]^v$, as it will be clear which v is meant. Specifically, we define $[\![E]\!]^v = \overline{v}(E)$ for all terms E in our data model.

Definition 4.9. Let α be a multiaction, with $\alpha = \langle a_1(\vec{d_1}), a_2(\vec{d_2}), \ldots, a_n(\vec{d_n}) \rangle$. The interpretation of a multiaction $[\![\alpha]\!]$ is defined by $[\![\langle a_1(\vec{d_1}), a_2(\vec{d_2}), \ldots, a_n(\vec{d_n}) \rangle]\!] = [a_1([\![\vec{d_1}]\!]), a_2([\![\vec{d_2}]\!]), \ldots, a_n([\![\vec{d_n}]\!])]$.

The following three definitions give a generic way to interpret the sets of the operators $\nabla, \partial, \Gamma, \tau, \rho$, described by V, IH, C, R in our syntax.

Definition 4.10. Let S be some syntactic set of terms (i.e. a collection of syntactic terms $t_1, \dots, t_n$ separated by commas and enclosed in { and }). The interpretation of S (as a true set) is defined as follows:

$$[\![\{t_1, \dots, t_n\}]\!] = \{[\![t_1]\!], \dots, [\![t_n]\!]\}$$

Definition 4.11. Let $t_1, t_2, \dots, t_n$ be some terms. The interpretation of $t_1|t_2|\dots|t_n$ is defined as follows:

$$[\![t_1|t_2|\dots|t_n]\!] = [[\![t_1]\!], [\![t_2]\!], \dots, [\![t_n]\!]]$$

Definition 4.12. Let t and t' be some syntactic terms. The interpretation of $\rightarrow$ is defined as follows:

$$[\![t \rightarrow t']\!] = \langle [\![t]\!], [\![t']\!] \rangle$$

In the semantics we use certain functions which are defined below. These functions are for instance needed to determine whether or not ∇_V will (dis)allow an action or whether or not an action is hidden by τ_I.

Definition 4.13. Let S be a function $\mathcal{N}_A \rightarrow \mathcal{N}_A$ and let $a(\overrightarrow{d}) \in \boldsymbol{A}$. Also, let $m \in \mathbb{B}(\boldsymbol{A})$. The *function mapping* operator $\bullet : (\mathcal{N}_A \rightarrow \mathcal{N}_A) \times \mathbb{B}(\boldsymbol{A}) \rightarrow \mathbb{B}(\boldsymbol{A})$ is defined as follows:

$$\begin{array}{lcl} S \bullet [] & = & [] \\ S \bullet ([a(\overrightarrow{d})] \oplus m) & = & [S^+(a)(\overrightarrow{d})] \oplus (S \bullet m) \end{array}$$

Definition 4.14. Let I be a set of action names (i.e. $I \subseteq \mathcal{N}_A$) and let $a(\overrightarrow{d}) \in \boldsymbol{A}$. Also, let $m \in \mathbb{B}(\boldsymbol{A})$. The *hiding* function $\theta : \mathbb{B}(\boldsymbol{A}) \times \mathcal{P}(\mathcal{N}_A) \rightarrow \mathbb{B}(\boldsymbol{A})$ is defined as follows:

$$\begin{array}{lcll} \theta([], I) & = & [] & \\ \theta([a(\overrightarrow{d})] \oplus m, I) & = & \theta(m, I) & \textit{if } a \in I \\ \theta([a(\overrightarrow{d})] \oplus m, I) & = & [a(\overrightarrow{d})] \oplus \theta(m, I) & \textit{if } a \notin I \end{array}$$

Definition 4.15. Let $a(\overrightarrow{d}) \in \boldsymbol{A}$ and $m \in \mathbb{B}(\boldsymbol{A})$. The *data stripping* function $\mu : \mathbb{B}(\boldsymbol{A}) \rightarrow \mathbb{B}(\mathcal{N}_A)$ is defined as follows:

$$\begin{array}{lcl} \mu([]) & = & [] \\ \mu([a(\overrightarrow{d})] \oplus m) & = & [a] \oplus \mu(m) \end{array}$$

For the communication operator we need a somewhat more complex definition. We introduce a communication function that takes the interpretation of a multiaction and finds all occurrences of left hand sides in the C parameter of the operator and replaces those occurrences with the corresponding right hand side. Note that this is only allowed if the data parameters are equal.

Definition 4.16. Let $m \in \mathbb{B}(\boldsymbol{A})$ and $a \in \mathcal{N}_A$. Also, let $\overrightarrow{d}, \overrightarrow{e} \in \overrightarrow{D_{\mathcal{M}}}$. The function $\chi : \mathbb{B}(\boldsymbol{A}) \times \overrightarrow{D_{\mathcal{M}}} \rightarrow B$ is true if, and only if, all actions of the multiaction parameter have the given data vector as parameter, i.e. χ is defined as follows:

$$\begin{array}{lcll} \chi([], \overrightarrow{d}) & = & t & \\ \chi([a(\overrightarrow{e})] \oplus m, \overrightarrow{d}) & = & \chi(m, \overrightarrow{d}) & \textit{if } \overrightarrow{d} = \overrightarrow{e} \\ \chi([a(\overrightarrow{e})] \oplus m, \overrightarrow{d}) & = & f & \textit{if } \overrightarrow{d} \neq \overrightarrow{e} \end{array}$$

Definition 4.17. Let $N_{\mathbb{B}} = \{n \mid n \in \mathbb{B}(\mathcal{N}_A) \wedge 1 < |n|\}$, $a(\overrightarrow{d}) \in \boldsymbol{A}$, $b \in N_{\mathbb{B}}$ and $m, n, o \in \mathbb{B}(\boldsymbol{A})$. Also let $C : N_{\mathbb{B}} \rightarrow (\mathcal{N}_A \cup \{\tau\})$ with $\forall_{\langle b,a \rangle, \langle c,a \rangle \in C}(\forall_{n \in b}(n \notin c))$. The *communication* function $\gamma : \mathbb{B}(\boldsymbol{A}) \times (N_{\mathbb{B}} \rightarrow (\mathcal{N}_A \cup \{\tau\})) \rightarrow \mathbb{B}(\boldsymbol{A})$ is defined by the following definition:

$$\begin{array}{lll} \gamma(m \oplus n, C) & = & [a(\vec{d})] \oplus \gamma(n, C) \quad \exists_{\langle b,a\rangle \in C}(b = \mu(m) \wedge \chi(m, \vec{d})) \\ \gamma(m \oplus n, C) & = & \gamma(n, C) \quad \exists_{\langle b,\tau\rangle \in C}(b = \mu(m) \wedge \chi(m, \vec{d})) \\ \gamma(m, C) & = & m \quad \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \\ & & \qquad \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))) \end{array}$$

Note that the extra condition on C is required to make γ a true function (i.e. $\gamma(m, C)$ is a unique bag). This is also a restriction on the syntax that was given earlier.

Definition 4.18. Let $t \in T_P$. Also, let $D \in \mathbb{D}$, $d \in V_D$ and let E be some expression of type D in which no variables occur that are bound in t. The *substitution* $t[E/d]$ means t with E substituted for all free occurrences of d in t.

Definition 4.19. Let X be a process variable and t a term (possibly containing X). A *recursive specification* is a set of equations $X(\overrightarrow{d{:}D}) = t$.

Let p and q be terms that do not contain any free variables and let $m \in \mathbb{B}(\mathcal{A})$. The relation $p \xrightarrow{m} q$ states that the process described by p can make a *transition* by executing a multiaction, with interpretation m, and will behave as the process described by q after it. The predicate $p \xrightarrow{m} \surd$ states that the process described by p can *terminate* by executing a multiaction with interpretation m.

Definition 4.20. Let $\mathcal{A} = \langle \Sigma, \mathcal{M} \rangle$ be a data algebra and $D_\mathcal{M} \in \mathcal{D}$ corresponding to type $D \in \mathbb{D}$. Also, let E be a set of recursive specifications. We define *process semantics* $Sem(\mathcal{A}, E) = \langle T_P, \mathbb{B}(\mathcal{A}), \longrightarrow, \longrightarrow \surd \rangle$, with a *transition* predicate $\longrightarrow$, taking two processes (from T_P) and an interpretation of an action (from $\mathbb{B}(\mathcal{A})$) as parameters, and a *termination* predicate $\longrightarrow \surd$, taking one process and an interpretation of an action as parameters, inductively by the following rules.

$$\frac{}{a(\vec{d}) \xrightarrow{[a([\![\vec{d}]\!])]} \surd} \qquad \frac{}{\alpha \xrightarrow{[\![\alpha]\!]} \surd} \qquad \frac{}{\tau \xrightarrow{[]} \surd}$$

$$\frac{t \xrightarrow{m} \surd}{\begin{array}{l} t + u \xrightarrow{m} \surd \\ u + t \xrightarrow{m} \surd \end{array}} \qquad \frac{t \xrightarrow{m} t'}{\begin{array}{l} t + u \xrightarrow{m} t' \\ u + t \xrightarrow{m} t' \end{array}}$$

$$\frac{t \xrightarrow{m} \surd}{t \cdot u \xrightarrow{m} u} \qquad \frac{t \xrightarrow{m} t'}{t \cdot u \xrightarrow{m} t' \cdot u}$$

$$\frac{t \xrightarrow{m} \surd}{b \rightarrow t \xrightarrow{m} \surd}\ \mathcal{A} \models b \qquad \frac{t \xrightarrow{m} t'}{b \rightarrow t \xrightarrow{m} t'}\ \mathcal{A} \models b$$

$$\frac{t[e/d] \xrightarrow{m} \surd}{\sum_{d:D} t \xrightarrow{m} \surd}\ e \in T_\Sigma \qquad \frac{t[e/d] \xrightarrow{m} t'}{\sum_{d:D} t \xrightarrow{m} t'}\ e \in T_\Sigma$$

$$\frac{t \xrightarrow{m} \surd}{\begin{array}{l} t \parallel u \xrightarrow{m} u \\ u \parallel t \xrightarrow{m} u \end{array}} \qquad \frac{t \xrightarrow{m} t'}{\begin{array}{l} t \parallel u \xrightarrow{m} t' \parallel u \\ u \parallel t \xrightarrow{m} u \parallel t' \end{array}} \qquad \frac{t \xrightarrow{m} \surd, u \xrightarrow{n} \surd}{t \parallel u \xrightarrow{m \oplus n} \surd}$$

$$\frac{t \xrightarrow{m} t', u \xrightarrow{n} u'}{t \parallel u \xrightarrow{m \oplus n} t' \parallel u'} \qquad \frac{t \xrightarrow{m} \surd, u \xrightarrow{n} u'}{\begin{array}{l} t \parallel u \xrightarrow{m \oplus n} u' \\ u \parallel t \xrightarrow{m \oplus n} u' \end{array}}$$

$$\frac{t \xrightarrow{m} \surd}{t \mathbin{\|\!\lfloor} u \xrightarrow{m} u} \qquad \frac{t \xrightarrow{m} t'}{t \mathbin{\|\!\lfloor} u \xrightarrow{m} t' \parallel u}$$

$$\frac{t \xrightarrow{m} \surd,\, u \xrightarrow{n} \surd}{t|u \xrightarrow{m\oplus n} \surd} \qquad \frac{t \xrightarrow{m} \surd,\, u \xrightarrow{n} u'}{\begin{array}{c} t|u \xrightarrow{m\oplus n} u' \\ u|t \xrightarrow{m\oplus n} u' \end{array}} \qquad \frac{t \xrightarrow{m} t',\, u \xrightarrow{n} u'}{t|u \xrightarrow{m\oplus n} t' \parallel u'}$$

$$\frac{t \xrightarrow{m} \surd}{\nabla_V(t) \xrightarrow{m} \surd}\ \mu(m) \in \llbracket V \rrbracket \cup \{[]\} \qquad \frac{t \xrightarrow{m} t'}{\nabla_V(t) \xrightarrow{m} \nabla_V(t')}\ \mu(m) \in \llbracket V \rrbracket \cup \{[]\}$$

$$\frac{t \xrightarrow{m} \surd}{\Gamma_C(t) \xrightarrow{\gamma(m,\llbracket C \rrbracket)} \surd} \qquad \frac{t \xrightarrow{m} t'}{\Gamma_C(t) \xrightarrow{\gamma(m,\llbracket C \rrbracket)} \Gamma_C(t')}$$

$$\frac{t \xrightarrow{m} \surd}{\partial_H(t) \xrightarrow{m} \surd}\ \mu(m) \cap \llbracket H \rrbracket = \emptyset \qquad \frac{t \xrightarrow{m} t'}{\partial_H(t) \xrightarrow{m} \partial_H(t')}\ \mu(m) \cap \llbracket H \rrbracket = \emptyset$$

$$\frac{t \xrightarrow{m} \surd}{\tau_I(t) \xrightarrow{\theta(m,\llbracket I \rrbracket)} \surd} \qquad \frac{t \xrightarrow{m} t'}{\tau_I(t) \xrightarrow{\theta(m,\llbracket I \rrbracket)} \tau_I(t')}$$

$$\frac{t \xrightarrow{m} \surd}{\rho_R(t) \xrightarrow{\llbracket R \rrbracket \bullet m} \surd} \qquad \frac{t \xrightarrow{m} t'}{\rho_R(t) \xrightarrow{\llbracket R \rrbracket \bullet m} \rho_R(t')}$$

$$\frac{t[\vec{e}/\vec{d}] \xrightarrow{m} \surd}{X(\vec{e}) \xrightarrow{m} \surd}\ \begin{array}{c} X(\overrightarrow{d:D}) = t \in E \\ \vec{e} \in \vec{T_\Sigma} \end{array} \qquad \frac{t[\vec{e}/\vec{d}] \xrightarrow{m} t'}{X(\vec{e}) \xrightarrow{m} t'}\ \begin{array}{c} X(\overrightarrow{d:D}) = t \in E \\ \vec{e} \in \vec{T_\Sigma} \end{array}$$

Table 1: GenSpect Semantics

To be able to compare and calculate with processes, we need to know when two processes are equal (i.e. have the same behaviour). We use the equality given by the following definition.

Definition 4.21. Let $G = Sem(\mathcal{A}, E) = \langle T_P, \mathbb{B}(A), \longrightarrow, \longrightarrow \surd \rangle$, with $\mathcal{A}$ a data algebra, E a set of process expressions and $\longrightarrow$ and $\longrightarrow \surd$ the transition relations on processes T_P with multiactions $\mathbb{B}(A)$, be a process semantics. Also, let t,t',u and u' be process terms in which process variables may occur only if they are in E and $m \in \mathbb{B}(A)$. A *bisimulation* is a relation B on processes such that if tBu:

- for all t' and m, $t \xrightarrow{m} t'$ means that there exists a u' with $u \xrightarrow{m} u'$ and $t'Bu'$
- for all u' and m, $u \xrightarrow{m} u'$ means that there exists a t' with $t \xrightarrow{m} t'$ and $t'Bu'$
- for all m, $t \xrightarrow{m} \surd$ means that $u \xrightarrow{m} \surd$
- for all m, $u \xrightarrow{m} \surd$ means that $t \xrightarrow{m} \surd$

We write the union of all bisimulation relations B as $\underline{\leftrightarrow}$. To state that two processes p and q are *bisimilar* we write $G \models p \underline{\leftrightarrow} q$ (or just $p \underline{\leftrightarrow} q$).

As the rules in Table 1 are in the *path* format[1], we have that bisimulation $\leftrightarrow$ is a congruence.

5 Axioms

To be able to calculate (more) easily, we introduce the following axiomatisation for the semantics given in the previous section, which is proved to be sound and complete in the next two sections. (Note that this axiomatisation does not include recursive process expressions and thus completeness only considers the semantics without these expressions.)

Definition 5.1. Let $t, u \in T_P$. We can *derive* t to u, notation $G \vdash t \doteq u$, when this follows from the rules and axioms below (and those of the data algebra). Usually we just write $p \doteq q$. The following rules hold for $\doteq$ (with $t, u, v \in T_P$, $x \in V_P$, $D \in \mathbb{D}$, $x_D, y_D \in V_D$, $t_d \in T_\Sigma$, $T_{\overline{\Sigma}}$ the set of closed data terms, $\star$ a unary process operator, $\diamond$ a binary operator with a data parameter and a process parameter and $\circ$ a binary process operator).

$$\frac{}{t \doteq t} \qquad \frac{t \doteq u}{\star(t) \doteq \star(u)}$$

$$\frac{t \doteq u}{u \doteq t} \qquad \frac{t_d =_d t'_d, u \doteq u'}{t_d \diamond u \doteq t'_d \diamond u'}$$

$$\frac{t \doteq u, u \doteq v}{t \doteq v} \qquad \frac{t \doteq t', u \doteq u'}{t \circ u \doteq t' \circ u'}$$

$$\frac{t \doteq u}{t[v/x] \doteq u[v/x]} \qquad \frac{\forall_{D \in \mathbb{D}}(\forall_{x_D \in V_D}(\forall_{e \in T_{\overline{\Sigma}}}(t[e/x_D] \doteq u[e/x_D])))}{t \doteq u}$$

$$\frac{t \doteq u}{t[E_D/x_D] \doteq u[E_D/x_D]} \qquad \frac{\forall_{x \in V_P}(\forall_{v \in T_{pc}}(t[v/x] \doteq u[v/x]))}{t \doteq u}$$

$$\frac{}{\sum_{x_D:D} t \doteq \sum_{y_D:D} t[y_D/x_D]} \; y_D \text{ does not occur in } t$$

Our axiomatisation is the following:

MA1	$a \doteq \langle a \rangle$	*CD1*	$\delta \vert \alpha \doteq \delta$
MA2	$\tau \doteq \langle\rangle$	*CD2*	$\alpha \vert \delta \doteq \delta$
MA3	$\langle \vec{a}, \vec{b} \rangle \doteq \langle \vec{b}, \vec{a} \rangle$		
		VD	$\nabla_V(\delta) \doteq \delta$
A1	$x + y \doteq y + x$	*V1*	$\nabla_V(\alpha) \doteq \alpha$ *if* $\mu([\![\alpha]\!]) \in [\![V]\!] \cup \{[]\}$
A2	$x + (y + z) \doteq (x + y) + z$	*V2*	$\nabla_V(\alpha) \doteq \delta$ *if* $\mu([\![\alpha]\!]) \notin [\![V]\!] \cup \{[]\}$
A3	$x + x \doteq x$	*V3*	$\nabla_V(x + y) \doteq \nabla_V(x) + \nabla_V(y)$
A4	$(x + y) \cdot z \doteq x \cdot z + y \cdot z$	*V4*	$\nabla_V(x \cdot y) \doteq \nabla_V(x) \cdot \nabla_V(y)$
A5	$(x \cdot y) \cdot z \doteq x \cdot (y \cdot z)$	*V6*	$\nabla_V(\sum_{d:D} p) \doteq \sum_{d:D}(\nabla_V(p))$
A6	$x + \delta \doteq x$		
A7	$\delta \cdot x \doteq \delta$	*DD*	$\partial_H(\delta) \doteq \delta$
		D1	$\partial_H(\alpha) \doteq \alpha$ *if* $\mu([\![\alpha]\!]) \cap [\![H]\!] = \emptyset$
C1	$\mathbf{t} \rightarrow x \doteq x$	*D2*	$\partial_H(\alpha) \doteq \delta$ *if* $\mu([\![\alpha]\!]) \cap [\![H]\!] \neq \emptyset$
C2	$\mathbf{f} \rightarrow x \doteq \delta$	*D3*	$\partial_H(x + y) \doteq \partial_H(x) + \partial_H(y)$

		$D4$	$\partial_H(x \cdot y) \doteq \partial_H(x) \cdot \partial_H(y)$
$SUM1$	$\sum_{d:D} x \doteq x$	$D6$	$\partial_H(\sum_{d:D} p) \doteq \sum_{d:D}(\partial_H(p))$
$SUM3$	$\sum_{d:D} p \doteq \sum_{d:D} p + p[e/d]$ *with* $e \in D$		
$SUM4$	$\sum_{d:D}(p+q) \doteq \sum_{d:D} p + \sum_{d:D} q$	TID	$\tau_I(\delta) \doteq \delta$
$SUM5$	$(\sum_{d:D} p) \cdot y \doteq \sum_{d:D}(p \cdot y)$	$TI1$	$\tau_I(\alpha) \doteq \beta$ *with* $[\![\beta]\!] = \theta([\![\alpha]\!], I)$
$SUM6$	$(\sum_{d:D} p) \lfloor\!\lfloor y \doteq \sum_{d:D}(p \lfloor\!\lfloor y)$	$TI3$	$\tau_I(x+y) \doteq \tau_I(x) + \tau_I(y)$
$SUM7$	$(\sum_{d:D} p) \mid y \doteq \sum_{d:D}(p \mid y)$	$TI4$	$\tau_I(x \cdot y) \doteq \tau_I(x) \cdot \tau_I(y)$
$SUM7'$	$x \mid (\sum_{d:D} q) \doteq \sum_{d:D}(x \mid q)$	$TI6$	$\tau_I(\sum_{d:D} p) \doteq \sum_{d:D}(\tau_I(p))$
$CM1$	$x \parallel y \doteq x \lfloor\!\lfloor y + y \lfloor\!\lfloor x + x \mid y$	RD	$\rho_R(\delta) \doteq \delta$
$CM2$	$\boldsymbol{\alpha} \lfloor\!\lfloor x \doteq \boldsymbol{\alpha} \cdot x$	$R1$	$\rho_R(\alpha) \doteq \beta$ *with* $[\![\beta]\!] = [\![R]\!] \bullet [\![\alpha]\!]$
$CM3$	$\boldsymbol{\alpha} \cdot x \lfloor\!\lfloor y \doteq \boldsymbol{\alpha} \cdot (x \parallel y)$	$R3$	$\rho_R(x+y) \doteq \rho_R(x) + \rho_R(y)$
$CM4$	$(x+y) \lfloor\!\lfloor z \doteq x \lfloor\!\lfloor z + y \lfloor\!\lfloor z$	$R4$	$\rho_R(x \cdot y) \doteq \rho_R(x) \cdot \rho_R(y)$
$CM5$	$\boldsymbol{\alpha} \cdot x \mid \boldsymbol{\beta} \doteq (\boldsymbol{\alpha} \mid \boldsymbol{\beta}) \cdot x$	$R6$	$\rho_R(\sum_{d:D} p) \doteq \sum_{d:D}(\rho_R(p))$
$CM6$	$\boldsymbol{\alpha} \mid \boldsymbol{\beta} \cdot x \doteq (\boldsymbol{\alpha} \mid \boldsymbol{\beta}) \cdot x$		
$CM7$	$\boldsymbol{\alpha} \cdot x \mid \boldsymbol{\beta} \cdot y \doteq (\boldsymbol{\alpha} \mid \boldsymbol{\beta}) \cdot (x \parallel y)$	GD	$\Gamma_C(\delta) \doteq \delta$
$CM8$	$(x+y) \mid z \doteq x \mid z + y \mid z$	$G1$	$\Gamma_C(\alpha) \doteq \beta$ *with* $[\![\beta]\!] = \gamma([\![\alpha]\!], [\![C]\!])$
$CM9$	$x \mid (y+z) \doteq x \mid y + x \mid z$	$G3$	$\Gamma_C(x+y) \doteq \Gamma_C(x) + \Gamma_C(y)$
$CM10$	$\langle \vec{a} \rangle \mid \langle \vec{b} \rangle \doteq \langle \vec{a}, \vec{b} \rangle$	$G4$	$\Gamma_C(x \cdot y) \doteq \Gamma_C(x) \cdot \Gamma_C(y)$
		$G6$	$\Gamma_C(\sum_{d:D} p) \doteq \sum_{d:D}(\Gamma_C(p))$

With $a, b \in A$, $\alpha, \beta \in \mathbb{A}$, $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{A} \cup \{\delta\}$, $x, y \in V_P$ and $p, q \in T_{pc}$.

Table 2: GenSpect Axioms

When we do wish to use recursive processes, we will (at least) need RDP^- and RSP [2] like principles. We informally define them as follows.

Definition 5.2. We call a specification *guarded* if every occurrence of a process variable on the right hand side of an equation is preceded by at least one action (possibly after substitution).

Definition 5.3. A *solution* of a specification $X(\vec{d}:\vec{D}) = t$ is a function $p \in \overrightarrow{D_{\mathcal{M}}} \rightarrow T_{pc}$ such that $p(\vec{d}) \doteq t[p/X]$ holds, where $(X(\vec{e}))[p/X] = p(\vec{e})$.

Definition 5.4. The *Restricted Recursive Definition Principle* (RDP) states that every guarded specification has a solution.

Definition 5.5. The *Recursive Specification Principle* (RSP) states that every guarded specification has at most one solution.

6 Soundness of the axioms

The soundness of the axiomatisation in the previous section is stated in the following theorem.

Theorem 6.1. Let $p, q \in T_{pc}$. The axiomatisation of GenSpect is sound (i.e. $G \vdash p \doteq q \Rightarrow G \models p \underline{\leftrightarrow} q$).

Proof 6.1. In Appendix A the soundness of the axioms given above has been proved (one axiom per subsection). The structure of each of these proofs is the same, namely:

- A relation R is given for the axiom. For an axiom $t \doteq u$, the relation R is the minimal relation that satisfies $tRu \wedge pRp \wedge (qRr \Leftrightarrow rRq) \wedge E$, for all $p, q, r \in T_{pc}$ and for all instantiations of process variables in t and u. E is *true*, unless it is explicitly defined otherwise in the subsection.

- For each conjunct of the definition of R it is shown that the properties that define a bisimulation hold.

It is clear that if this is done, R is proven to be a bisimulation relation.

To shorten the proofs and get rid of trivial facts, that would otherwise be repeated in every subsection, we do not define R explicitly (as it is always of the form given above) or prove the conjuncts pRp and $qRr \Leftrightarrow rRq$ (and E, if it is *true*). The proofs of these conjuncts are trivial and therefore not given at all.

When we say a process cannot terminate in any of the subsections, we mean that there is no m for which $\xrightarrow{m} \surd$ holds for that process. (So it could very well be that the process can indeed terminate, but not in one "step".) □

Note that we should also prove the given rules of $\doteq$, but as these are quite straightforward (with, for example, the symmetry of $\leftrightarrow$ and the fact that $\leftrightarrow$ is a congruence) they are not given.

7 Completeness

To prove that our axiomatisation is complete (i.e. that whenever two processes p and q are bisimilar we know that $p \doteq q$) we introduce basic terms and show that any (process-closed) term can be rewritten to such a basic term. By doing this, the actual completeness proof only has to consider these basic terms.

Note that we often say that something holds "by induction", with which we mean to say that it follows from the (implicit) induction hypothesis. Also note that we use certain properties of $\rightarrow$ which, with induction on booleans, are trivially true and therefore no proof of them is given. The following properties are used (with b a boolean expression without variable d and $\star$ the unary process operators):

$$\begin{array}{ll}
b \rightarrow (x+y) \doteq b \rightarrow x + b \rightarrow y & b \rightarrow x + c \rightarrow x \doteq b \vee c \rightarrow x \\
(b \rightarrow x) \cdot y \doteq b \rightarrow (x \cdot y) & b \rightarrow c \rightarrow x \doteq b \wedge c \rightarrow x \\
\sum_{d:D} b \rightarrow x \doteq b \rightarrow \sum_{d:D} x & \star(b \rightarrow x) \doteq b \rightarrow \star(x) \\
(b \rightarrow x)|y \doteq b \rightarrow (x|y) & x|(b \rightarrow y) \doteq b \rightarrow (x|y) \\
(b \rightarrow x) \mathbin{\|\!\!\_} y \doteq b \rightarrow (x \mathbin{\|\!\!\_} y) &
\end{array}$$

We do not allow variable d in b because it would otherwise mean that axiom $\sum_{d:D} b \rightarrow x \doteq b \rightarrow \sum_{d:D}$ is not sound. If d does occur in b we can still apply this axiom to $b \rightarrow \sum_{d:D}$ if we first apply α-conversion to the summation. We do this implicitly where needed in the following proofs.

7.1 Basic terms

Definition 7.1. Let $p, q, r \in T_{pc}$ and let $\alpha \in \mathbb{A} \cup \{\delta\}$. Also let b be a data term of type B. A *basic term* p is a term having one of the following forms (we write $\overrightarrow{\sum}$ for $\sum_{d_1:D_1} \sum_{d_2:D_2} \cdots \sum_{d_n:D_n}$, with $n \geq 0$):

- $\overrightarrow{\sum} b \rightarrow \alpha$
- $\overrightarrow{\sum} b \rightarrow \alpha \cdot q$, with q a basic term
- $q + r$, with q and r basic terms

Definition 7.2. We call the operators that can occur in basic terms ($\cdot$, $+$, $\rightarrow$, $\sum$) *basic operators.*

Theorem 7.3. *Let $p, q \in T_{pc}$. For each p, which uses only basic operators, there exists a basic term q with $p \doteq q$.*

Proof 7.3. We use induction on the structure of p:

- $p = a(\overrightarrow{d})$, which means that $p = a(\overrightarrow{d}) \doteq \langle a(\overrightarrow{d})\rangle \doteq \boldsymbol{t} \rightarrow \langle a(\overrightarrow{d})\rangle$.
- $p = \alpha$, which means that $p = \alpha \doteq \boldsymbol{t} \rightarrow \alpha$.
- $p = \delta$, which means that $p = \delta \doteq \boldsymbol{t} \rightarrow \delta$.
- $p = \tau$, which means that $p = \tau \doteq \langle\rangle \doteq \boldsymbol{t} \rightarrow \langle\rangle$.
- $p = q + r$, which means that by induction we have basic terms q' and r' with $q \doteq q' \wedge r \doteq r'$, and $p = q + r \doteq q' + r'$.
- $p = q \cdot r$, which means that by induction we have basic terms q' and r' with $q \doteq q' \wedge r \doteq r'$. We now prove that for each basic term s there is a basic term t with $t \doteq s \cdot r$, which means that there is a basic term p' with $p = q \cdot r \doteq q' \cdot r \doteq p'$. By the structure of s:
  - $\overrightarrow{\sum} b \rightarrow \alpha$, which means that $s \cdot r = (\overrightarrow{\sum} b \rightarrow \alpha) \cdot r \doteq \overrightarrow{\sum}(b \rightarrow \alpha) \cdot r \doteq \overrightarrow{\sum} b \rightarrow \alpha \cdot r \doteq \overrightarrow{\sum} b \rightarrow \alpha \cdot r'$, or
  - $\overrightarrow{\sum} b \rightarrow \alpha \cdot s'$, with s' a basic term, which means that $s \cdot r = (\overrightarrow{\sum} b \rightarrow \alpha \cdot s') \cdot r \doteq \overrightarrow{\sum}(b \rightarrow \alpha \cdot s') \cdot r \doteq \overrightarrow{\sum} b \rightarrow \alpha \cdot s' \cdot r \doteq \overrightarrow{\sum} b \rightarrow \alpha \cdot u$, with basic term $u \doteq s' \cdot r$ by induction, or
  - $s' + s''$, with s' and s'' basic terms, which means that $s \cdot r = (s' + s'') \cdot r \doteq s' \cdot r + s'' \cdot r \doteq u + u'$, with basic terms $u \doteq s' \cdot r$ and $u' \doteq s'' \cdot r$ by induction.
- $p = b \rightarrow q$, which means that by induction we have a basic term q' with $q \doteq q'$, and the structure of q' is:
  - $\overrightarrow{\sum} c \rightarrow \alpha$, which means that $p = b \rightarrow q \doteq b \rightarrow q' = b \rightarrow (\overrightarrow{\sum} c \rightarrow \alpha) \doteq \overrightarrow{\sum} b \rightarrow c \rightarrow \alpha \doteq \overrightarrow{\sum} b \wedge c \rightarrow \alpha$
  - $\overrightarrow{\sum} c \rightarrow \alpha \cdot r$, with r a basic term, which means that $p = b \rightarrow q \doteq b \rightarrow q' = b \rightarrow (\overrightarrow{\sum} c \rightarrow \alpha \cdot r) \doteq \overrightarrow{\sum} b \rightarrow c \rightarrow \alpha \cdot r \doteq \overrightarrow{\sum} b \wedge c \rightarrow \alpha \cdot r$
  - $r + r'$, with r and r' basic terms, which means that $p = b \rightarrow q \doteq b \rightarrow q' = b \rightarrow (r + r') \doteq b \rightarrow r + b \rightarrow r' \doteq s + s'$, with basic terms $s \doteq b \rightarrow r$ and $s' \doteq b \rightarrow r'$ by induction.
- $p = \sum_{d:D} q$, which means that by induction we have a basic term q' with $q \doteq q'$. We now use induction on q' to prove that there is a basic term r with $r \doteq \sum_{d:D} q'$, from which follows that there is a basic term r' with $p = \sum_{d:D} q \doteq \sum_{d:D} q' \doteq r'$. If q' is of one of the first two forms, $\sum_{d:D} q'$ is trivially a basic term (as the $\sum_{d:D}$ simply becomes part of the $\overrightarrow{\sum}$ of q'). In the third case we have $q' = s + s'$ with basic terms s and s', which means that $\sum_{d:D} q' = \sum_{d:D}(s + s') \doteq \sum_{d:D} s + \sum_{d:D} s' \doteq t + t'$, with basic terms $t \doteq \sum_{d:D} s$ and $t' \doteq \sum_{d:D} s'$ by induction.

□

Lemma 7.4. Let $p, q \in T_{pc}$. If there are no non-basic operators in p and q, then there is a term $r \in T_{pc}$ without non-basic operators, such that $r \doteq p \circ q$ (for some non-basic binary operator $\circ$) or $r \doteq \star(p)$ (for some non-basic unary operator $\star$) .

Proof 7.4. By Theorem 7.3 we know that there are basic terms p' and q' with $p \doteq p'$ and $q \doteq q'$. We prove, by induction on the number of symbols in p' and q', that there is a term r' without non-basic operators, such that $r' \doteq p' \circ q'$ (or $r' \doteq \star(p')$) and thus $r' \doteq p \circ q$ (or $r' \doteq \star(p)$).

- $p' \mathbin{|\!\lfloor} q'$, and p' is of structure

    - $\overrightarrow{\sum} b \rightarrow \alpha$, which means that $= p' \mathbin{\lfloor\!\lfloor} q' = (\overrightarrow{\sum} b \rightarrow \alpha) \mathbin{\lfloor\!\lfloor} q' \doteq \overrightarrow{\sum}(b \rightarrow \alpha) \mathbin{\lfloor\!\lfloor} q' \doteq \overrightarrow{\sum} b \rightarrow (\alpha \mathbin{\lfloor\!\lfloor} q') \doteq \overrightarrow{\sum} b \rightarrow \alpha \cdot q'$, or
    - $\overrightarrow{\sum} b \rightarrow \alpha \cdot s$, with s a basic term, which means that $p' \mathbin{\lfloor\!\lfloor} q' = (\overrightarrow{\sum} b \rightarrow \alpha \cdot s) \mathbin{\lfloor\!\lfloor} q' \doteq \overrightarrow{\sum}(b \rightarrow \alpha \cdot s) \mathbin{\lfloor\!\lfloor} q' \doteq \overrightarrow{\sum} b \rightarrow (\alpha \cdot s \mathbin{\lfloor\!\lfloor} q') \doteq \overrightarrow{\sum} b \rightarrow \alpha \cdot (s \parallel q') = \overrightarrow{\sum} b \rightarrow \alpha \cdot s'$, with $s' \doteq s \parallel q'$ a term without non-basic operators by induction, or
    - $s + s'$, with s and s' basic terms, which means that $p' \mathbin{\lfloor\!\lfloor} q' = (s + s') \mathbin{\lfloor\!\lfloor} q' \doteq s \mathbin{\lfloor\!\lfloor} q' + s' \mathbin{\lfloor\!\lfloor} q' = t + t'$, with $t \doteq s \mathbin{\lfloor\!\lfloor} q'$ and $t' \doteq s' \mathbin{\lfloor\!\lfloor} q'$ terms without non-basic operators by induction.

- $p'|q'$, and

    - $p' = \overrightarrow{\sum} b \rightarrow \alpha \wedge q' = \overrightarrow{\sum} b' \rightarrow \alpha'$, which means that $p'|q' = (\overrightarrow{\sum} b \rightarrow \alpha)|(\overrightarrow{\sum} b' \rightarrow \alpha') \doteq \overrightarrow{\sum}((b \rightarrow \alpha)|(\overrightarrow{\sum} b' \rightarrow \alpha')) \doteq \overrightarrow{\sum}\overrightarrow{\sum}((b \rightarrow \alpha)|(b' \rightarrow \alpha')) \doteq \overrightarrow{\sum} b \rightarrow (\alpha|(b' \rightarrow \alpha')) \doteq \overrightarrow{\sum} b \rightarrow b' \rightarrow (\alpha|\alpha') \doteq \overrightarrow{\sum} b \wedge b' \rightarrow (\alpha|\alpha')$, with α and α' actions or δ, which means that $\alpha|\alpha'$ is an action or δ
    - $p' = \overrightarrow{\sum} b \rightarrow \alpha \cdot s \wedge q' = \overrightarrow{\sum} b' \rightarrow \alpha'$, with s a basic term, which means that $p'|q' = (\overrightarrow{\sum} b \rightarrow \alpha \cdot s)|(\overrightarrow{\sum} b' \rightarrow \alpha') \doteq \overrightarrow{\sum}((b \rightarrow \alpha \cdot s)|(\overrightarrow{\sum} b' \rightarrow \alpha')) \doteq \overrightarrow{\sum}\overrightarrow{\sum}((b \rightarrow \alpha \cdot s)|(b' \rightarrow \alpha')) \doteq \overrightarrow{\sum} b \rightarrow (\alpha \cdot s|(b' \rightarrow \alpha')) \doteq \overrightarrow{\sum} b \rightarrow b' \rightarrow (\alpha \cdot s|\alpha') \doteq \overrightarrow{\sum} b \wedge b' \rightarrow (\alpha|\alpha') \cdot s$, with α and α' actions or δ, which means that $\alpha|\alpha'$ is an action or δ
    - $p' = \overrightarrow{\sum} b \rightarrow \alpha \wedge q' = \overrightarrow{\sum} b' \rightarrow \alpha' \cdot s$, with s a basic term, which is symmetrical to the previous case
    - $p' = \overrightarrow{\sum} b \rightarrow \alpha \cdot s \wedge q' = \overrightarrow{\sum} b' \rightarrow \alpha' \cdot s'$, with s and s' basic terms, which means that $p'|q' = (\overrightarrow{\sum} b \rightarrow \alpha \cdot s)|(\overrightarrow{\sum} b' \rightarrow \alpha' \cdot s') \doteq \overrightarrow{\sum}((b \rightarrow \alpha \cdot s)|(\overrightarrow{\sum} b \rightarrow \alpha' \cdot s')) \doteq \overrightarrow{\sum}\overrightarrow{\sum}((b \rightarrow \alpha \cdot s)|(b' \rightarrow \alpha' \cdot s')) \doteq \overrightarrow{\sum} b \rightarrow (\alpha \cdot s|(b' \rightarrow \alpha' \cdot s')) \doteq \overrightarrow{\sum} b \rightarrow b' \rightarrow (\alpha \cdot s|\alpha' \cdot s') \doteq \overrightarrow{\sum} b \wedge b' \rightarrow (\alpha|\alpha') \cdot (s \parallel s') = \overrightarrow{\sum} b \wedge b' \rightarrow (\alpha|\alpha') \cdot t$, with α and α' actions or δ, which means that $\alpha|\alpha'$ is an action or δ, and $t \doteq s \parallel s'$ a term without non-basic operators by induction
    - $p' = s+s'$, with basic terms s and s', which means that $p'|q' = (s+s')|q' \doteq s|q'+s'|q' = t+t'$, with $t \doteq s|q'$ and $t' \doteq s'|q'$ terms without non-basic operators by induction
    - $q' = s + s'$, with basic terms s and s', which is symmetrical to the previous case.

- $p' \parallel q'$, which means that $p' \parallel q' \doteq p' \mathbin{\lfloor\!\lfloor} q' + q' \mathbin{\lfloor\!\lfloor} p' + p'|q' = s + t + u$, with $s \doteq p' \mathbin{\lfloor\!\lfloor} q'$, $t \doteq q' \mathbin{\lfloor\!\lfloor} p'$ and $u \doteq p'|q'$ terms without non-basic operators by the previous cases.

- $\nabla_V(p')$, and p' is of structure

    - $\overrightarrow{\sum} b \rightarrow \alpha$, which means that $\nabla_V(p') \doteq \nabla_V(\overrightarrow{\sum} b \rightarrow \alpha) \doteq \overrightarrow{\sum}(\nabla_V(b \rightarrow \alpha)) \doteq \overrightarrow{\sum} b \rightarrow \nabla_V(\alpha)$, with $\nabla_V(\alpha)$ an action or δ, or
    - $\overrightarrow{\sum} b \rightarrow \alpha \cdot s$, with s a basic term, which means that $\nabla_V(p') = \nabla_V(\overrightarrow{\sum} b \rightarrow \alpha \cdot s) \doteq \overrightarrow{\sum} \nabla_V(b \rightarrow \alpha \cdot s) \doteq \overrightarrow{\sum} b \rightarrow \nabla_V(\alpha \cdot s) \doteq \overrightarrow{\sum} b \rightarrow \nabla_V(\alpha) \cdot \nabla_V(s) = \overrightarrow{\sum} b \rightarrow \nabla_V(\alpha) \cdot s'$, with $\nabla_V(\alpha)$ an action or δ and $s' \doteq \nabla_V(s)$ a term without non-basic operators by induction, or
    - $s+s'$, with s and s' basic terms, which means that $\nabla_V(p') = \nabla_V(s+s') \doteq \nabla_V(s)+\nabla_V(s') = t + t'$, with $t \doteq \nabla_V(s)$ and $t' \doteq \nabla_V(s')$ terms without non-basic operators by induction.

- $\tau_I(p')$, which is similar to the previous case.

- $\rho_R(p')$, which is similar to the previous case.

- $\Gamma_C(p')$, which is similar to the previous case.

- $\partial_H(p')$, which is similar to the previous case.

□

Theorem 7.5. Let $p, q \in T_{pc}$. The *elimination theorem* states that all non-basic operators ($\|$, $\lfloor\!\lfloor$, $|$, ∇_V, Γ_C, τ_I and ρ_R) can be eliminated. That is, for each term p there exists a term q with $p \doteq q$ and q does not contain any non-basic operator.

Proof 7.5. With induction on the number of symbols in p:

- p is an action, deadlock or tau, which means there are no non-basic operators in p.
- $p = b \rightarrow q$, which means that we have a term q' with $q \doteq q'$ which contains no non-basic operators by induction and thus $p = b \rightarrow q \doteq b \rightarrow q'$ does not as well.
- $p = q \circ r$, with $\circ$ a basic operator (excluding $\rightarrow$), which means that we have terms q' and r' with $q \doteq q'$ and $r \doteq r'$ which contain no non-basic operators by induction and thus $p = q \circ r \doteq q' \circ r'$ does not as well.
- $p = \star(q)$, with $\star$ a non-basic unary operator, which means that we have a term q' with $q \doteq q'$ which contains no non-basic operators by induction and thus $p = \star(q) \doteq \star(q') \doteq r$, with r a term without non-basic operators by Lemma 7.4.
- $p = q \circ r$, with $\circ$ a non-basic binary operator, which means that we have terms q' and r' with $q \doteq q'$ and $r \doteq r'$ which contain no non-basic operators by induction and thus $p = q \circ r \doteq q' \circ r' \doteq s$, with s a term without non-basic operators by Lemma 7.4.

□

Theorem 7.6. Let $p, q \in T_{pr}$. For each p there exists a basic term q with $p \doteq q$.

Proof 7.6. By Theorem 7.5 we know there exists a r, with $p \doteq r$, in which no non-basic operators occur. And Theorem 7.3 states that for such a term there is a basic term q with $r \doteq q$. Thus, as $p \doteq r \doteq q$, there is a basic term for each process-closed term p. □

7.2 Relative completeness

As proved in [8], the data algebra needs to be complete and have equality on the data and quantifier elimination to be able to have completeness. This is expressed by the following definitions. (See [8] pp. 83-85 for the precise details.)

Definition 7.7. Let $\mathcal{A}$ be a data algebra. We say that $\mathcal{A}$ is complete if, and only if, $\overline{v}(t) = \overline{v}(u) \Rightarrow t \doteq_d u$ (for all closed data terms t and u and all valuations v).

Definition 7.8. Let $\mathcal{A}$ be a data algebra. We say that $\mathcal{A}$ has equality if, and only if, for every data type $D \in \mathbb{D}$ there is a function eq_D with specification $\langle eq_D, D, D, B\rangle$ for which the following holds (with d and e of type D and for all valuations v):

$$\begin{array}{ll} eq_D(d,e) = \boldsymbol{t} & \textit{if } \overline{v}(d) = \overline{v}(e) \\ eq_D(d,e) = \boldsymbol{f} & \textit{if } \overline{v}(d) \neq \overline{v}(e) \end{array}$$

To be able to use this equality we use the following additional axiom:

$$EQ \quad eq_{D_1}(d_1,e_1) \wedge \ldots \wedge eq_{D_n}(d_n,e_n) \rightarrow \alpha(d_1,\ldots,d_n) \doteq eq_{D_1}(d_1,e_1) \wedge \ldots \wedge eq_{D_n}(d_n,e_n) \rightarrow \alpha(e_1,\ldots,e_n)$$

Definition 7.9. Let $\mathcal{A}$ be a data algebra. We say that $\mathcal{A}$ has quantifier elimination if, and only if, for every process $p \in T_{pc}$ and boolean expression b depending on a variable d that does not occur in p, there exists a boolean expression c such that the following axiom holds:

$$QE \quad \textstyle\sum_{d:D} b \rightarrow p \doteq c \rightarrow p$$

Before we can prove completeness we need the following definitions and lemmas. Note that we use standard predicate calculus (with $\equiv, \Rightarrow, \wedge, \vee, \neg, \forall, \exists$) to define, derive and prove. This means we consider $p \underline{\leftrightarrow} q$ and $p \doteq q$ to be predicates on p and q and will write derivations like $p \doteq \alpha + \alpha \equiv p \doteq \alpha$, for some $p \in T_{pc}$. For $\mathcal{A} \models E$ we also just write E if no confusion can arise.

Definition 7.10. Let $p, q \in T_{pc}$. The *bisimulation inclusion* $\underline{\rightarrow}$ of a bisimulation $\underline{\leftrightarrow}$ is defined by $p \underline{\rightarrow} q \equiv p + q \underline{\leftrightarrow} q$.

Lemma 7.11. We can formulate the definition of $\underline{\rightarrow}$ as follows:

$$p \underline{\rightarrow} q \equiv \forall_{\alpha \in \mathbb{B}(A)}((p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q')))$$

Proof 7.11. With the definition of $\underline{\leftrightarrow}$ in the more formal form of $p \underline{\leftrightarrow} q \equiv \forall_{\alpha \in \mathbb{B}(A)}((p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge (q \xrightarrow{\alpha} \checkmark \Rightarrow p \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q')) \wedge \forall_{q'}(q \xrightarrow{\alpha} q' \Rightarrow \exists_{p'}(p \xrightarrow{\alpha} p' \wedge p' \underline{\leftrightarrow} q')))$ the proof is as follows:

$$\begin{array}{ll}
 & p \underline{\rightarrow} q \\
\equiv & p + q \underline{\leftrightarrow} q \\
\equiv & \forall_{\alpha \in \mathbb{B}(A)}((p + q \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge (q \xrightarrow{\alpha} \checkmark \Rightarrow p + q \xrightarrow{\alpha} \checkmark) \wedge \\
 & \quad \forall_{p'}(p + q \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q')) \wedge \forall_{q'}(q \xrightarrow{\alpha} q' \Rightarrow \exists_{p'}(p + q \xrightarrow{\alpha} p' \wedge p' \underline{\leftrightarrow} q'))) \\
\equiv & \forall_{\alpha \in \mathbb{B}(A)}(((p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge (q \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark)) \wedge true \wedge \\
 & \quad \forall_{p'}((p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q')) \wedge (q \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q'))) \wedge true) \\
\equiv & \forall_{\alpha \in \mathbb{B}(A)}(((p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge true) \wedge \forall_{p'}((p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q')) \wedge true)) \\
\equiv & \forall_{\alpha \in \mathbb{B}(A)}((p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \underline{\leftrightarrow} q')))
\end{array}$$

□

Definition 7.12. Let $p, q \in T_{pc}$. The *axiomatic inclusion* $\dot{\leqslant}$ of a axiomatic relation $\doteq$ is defined by $p \dot{\leqslant} q \equiv p + q \doteq q$.

Lemma 7.13. An axiomatic inclusion $\dot{\leqslant}$ is sound with respect to a bisimulation inclusion $\underline{\rightarrow}$ if $\doteq$ is sound with respect to $\underline{\leftrightarrow}$ (i.e. $\forall_{p,q}(p \doteq q \Rightarrow p \underline{\leftrightarrow} q) \Rightarrow \forall_{p,q}(p \dot{\leqslant} q \Rightarrow p \underline{\rightarrow} q)$).

Proof 7.13. Trivial. □

Lemma 7.14. Let $p, q \in T_{pc}$. The relation $\dot{\leqslant}$ is antisymmetric:

$$p \doteq q \equiv p \dot{\leqslant} q \wedge q \dot{\leqslant} p$$

Proof 7.14.

$$\begin{array}{ll}
 & p \dot{\leqslant} q \wedge q \dot{\leqslant} p \\
\equiv & p + q \doteq q \wedge q + p \doteq p \\
\equiv & q \doteq p + q \wedge p + q \doteq p \\
\Rightarrow & p \doteq q
\end{array}
\qquad
\begin{array}{ll}
 & p \doteq q \\
\equiv & p \doteq q \wedge p \doteq q \\
\equiv & p \doteq q \wedge p + p \doteq q \wedge q \doteq p \\
\equiv & p \doteq q \wedge p + q \doteq q \wedge q + q \doteq p \\
\equiv & p \doteq q \wedge p \dot{\leqslant} q \wedge q + p \doteq p \\
\Rightarrow & p \dot{\leqslant} q \wedge q \dot{\leqslant} p
\end{array}$$

□

Lemma 7.15. Let $p, q, r \in T_{pc}$. The axiomatic inclusion distributes over the alternative composition as follows:

$$p + q \mathrel{\dot{\leqslant}} r \equiv p \mathrel{\dot{\leqslant}} r \wedge q \mathrel{\dot{\leqslant}} r$$

Proof 7.15.

$$\begin{aligned}
& p + q \mathrel{\dot{\leqslant}} r \\
\equiv\ & p + q + r \doteq r \\
\equiv\ & p + q + r \doteq r \wedge p + q + r \doteq r \\
\equiv\ & p + q + r \doteq r \wedge p + q + r \doteq r \wedge p + q + q + r \doteq r \\
\equiv\ & p + q + r \doteq r \wedge p + p + q + r \doteq r \wedge q + p + q + r \doteq r \\
\Rightarrow\ & p + r \doteq r \wedge q + r \doteq r \\
\equiv\ & p \mathrel{\dot{\leqslant}} r \wedge q \mathrel{\dot{\leqslant}} r
\end{aligned}$$

$$\begin{aligned}
& p \mathrel{\dot{\leqslant}} r \wedge q \mathrel{\dot{\leqslant}} r \\
\equiv\ & p + r \doteq r \wedge q + r \doteq r \\
\Rightarrow\ & p + q + r \doteq r \\
\equiv\ & p + q \mathrel{\dot{\leqslant}} r
\end{aligned}$$

□

Definition 7.16. Let $p, q \in T_{pc}$ and b a boolean expression. We call b a p-simulation condition of q if $(\mathcal{A} \models b) \equiv p \mathrel{\underline{\rightharpoonup}} q$. (Or, with data, $\forall_{\vec{d}}((\mathcal{A} \models b(\vec{d})) \equiv p(\vec{d}) \mathrel{\underline{\rightharpoonup}} q(\vec{d}))$.)

The following lemmas express the relation between simulation conditions and the conditions in basic terms.

Lemma 7.17. Let b be a boolean expression and $p, q \in T_{pc}$. The following holds:

$$b \rightarrow p \mathrel{\underline{\rightharpoonup}} q \equiv b \Rightarrow p \mathrel{\underline{\rightharpoonup}} q$$

Proof 7.17.

$$\begin{aligned}
& b \rightarrow p \mathrel{\underline{\rightharpoonup}} q \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((b \rightarrow p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \cdot \checkmark) \wedge \forall_{p'}(b \rightarrow p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((b \wedge p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(b \wedge p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((b \Rightarrow p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(b \Rightarrow p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((b \Rightarrow p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge (b \Rightarrow \forall_{p'}(p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q')))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}(b \Rightarrow (p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & b \Rightarrow \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((p \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & b \Rightarrow p \mathrel{\underline{\rightharpoonup}} q
\end{aligned}$$

□

Lemma 7.18. Let D a data type and d a variable of that type. Also, let $p, q \in T_{pc}$. The following holds:

$$\textstyle\sum_{d:D} p(d) \mathrel{\underline{\rightharpoonup}} q \equiv \forall_{d:D}(p(d) \mathrel{\underline{\rightharpoonup}} q)$$

Proof 7.18.

$$\begin{aligned}
& \textstyle\sum_{d:D} p(d) \mathrel{\underline{\rightharpoonup}} q \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((\textstyle\sum_{d:D} p(d) \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(\textstyle\sum_{d:D} p(d) \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((\exists_{d:D}(p(d) \xrightarrow{\alpha} \checkmark) \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(\exists_{d:D}(p(d) \xrightarrow{\alpha} p') \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((\neg\exists_{d:D}(p(d) \xrightarrow{\alpha} \checkmark) \vee q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(\neg\exists_{d:D}(p(d) \xrightarrow{\alpha} p') \vee \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((\forall_{d:D}(\neg(p(d) \xrightarrow{\alpha} \checkmark)) \vee q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(\forall_{d:D}(\neg(p(d) \xrightarrow{\alpha} p')) \vee \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q'))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}(\forall_{d:D}(\neg(p(d) \xrightarrow{\alpha} \checkmark) \vee q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(\forall_{d:D}(\neg(p(d) \xrightarrow{\alpha} p') \vee \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q')))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}(\forall_{d:D}(p(d) \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{d:D}(\forall_{p'}(p(d) \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q')))) \\
\equiv\ & \forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}(\forall_{d:D}((p(d) \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p(d) \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q')))) \\
\equiv\ & \forall_{d:D}(\forall_{\alpha \in \mathbb{B}(\boldsymbol{A})}((p(d) \xrightarrow{\alpha} \checkmark \Rightarrow q \xrightarrow{\alpha} \checkmark) \wedge \forall_{p'}(p(d) \xrightarrow{\alpha} p' \Rightarrow \exists_{q'}(q \xrightarrow{\alpha} q' \wedge p' \mathrel{\underline{\leftrightarrow}} q')))) \\
\equiv\ & \forall_{d:D}(p(d) \mathrel{\underline{\rightharpoonup}} q)
\end{aligned}$$

□

Lemma 7.19. Let b be a boolean expression and $\overrightarrow{d}$ a vector of data variables. Also, let $p, q \in T_{pc}$. The following holds:

$$\sum_{\overrightarrow{d}} b(\overrightarrow{d}) \rightarrow p(\overrightarrow{d}) \underline{\rightarrow} q \equiv \forall_{\overrightarrow{d}}(b(\overrightarrow{d}) \Rightarrow p(\overrightarrow{d}) \underline{\rightarrow} q)$$

Proof 7.19.

$$\sum_{\overrightarrow{d}} b(\overrightarrow{d}) \rightarrow p(\overrightarrow{d}) \underline{\rightarrow} q \equiv \forall_{d:D}(b(\overrightarrow{d}) \rightarrow p(\overrightarrow{d}) \underline{\rightarrow} q) \equiv \forall_{d:D}(b(\overrightarrow{d}) \Rightarrow p(\overrightarrow{d}) \underline{\rightarrow} q)$$

□

Lemma 7.20. Let b and c be boolean expressions and $\overrightarrow{d}$ and $\overrightarrow{e}$ vectors of data variables. Also, let $p, q \in T_{pc}$. The following holds:

$$\sum_{\overrightarrow{d}} b(\overrightarrow{d}) \rightarrow p(\overrightarrow{d}) \underline{\rightarrow} q \equiv \forall_{\overrightarrow{d}}(b(\overrightarrow{d}) \Rightarrow c(\overrightarrow{d})), \text{ with } c(\overrightarrow{d}) \text{ a } p(\overrightarrow{d})\text{-simulation of } q \text{ for all } \overrightarrow{d}$$

Proof 7.20. Trivial with Lemma 7.19. □

Theorem 7.21. Let $p, q \in T_{pc}$. If data algebra $\mathcal{A}$ is complete and has equality and quantifier elimination, there exists a p-simulation for q.

Proof 7.21. This theorem follows from Theorem 5.26 in [8]. □

We write $\# : T_{pc} \rightarrow \mathbb{N}$ for the number of (process) symbols in a term. Its definition is quite straightforward and will therefore not be given in an other way than saying that every symbol (operator or (in)action) is counted. Note that this does not include data operators or constants.

Lemma 7.22. Let $p, q, q', r, s \in T_{pc}$ and $\mathcal{A}$ be complete and have equality and quantifier elimination. We can split p in smaller (or equally large) pieces if $p \underline{\rightarrow} q + q'$, as expressed by the following:

$$p \underline{\rightarrow} q + q' \Rightarrow \exists_{r,s}(p \doteq r + s \wedge \#(r) \leq \#(p) \wedge \#(s) \leq \#(p) \wedge r \underline{\rightarrow} q \wedge s \underline{\rightarrow} q')$$

Proof 7.22. With induction on the structure of p:

- $p = \overrightarrow{\sum} b \rightarrow \alpha$. Let b_1 and b_2 be α-simulations for q and q' resp. (by Theorem 7.21) and define $p_i = \overrightarrow{\sum} b \wedge b_i \rightarrow \alpha$, with $i \in \{1, 2\}$. We need to show that there exist r and s such that (i) $p \doteq r + s$, (ii) $\#(r) \leq \#(p) \wedge \#(s) \leq \#(p)$ and (iii) $r \underline{\rightarrow} q \wedge s \underline{\rightarrow} q'$. By choosing p_1 and p_2 for r and s, (ii) holds trivially and (iii) holds by Lemma 7.20. We now only need to show that $p \doteq p_1 + p_2$. Observe that $b \wedge (b_1 \vee b_2)$ is a α-simulation condition for $q + q'$ by Lemma 7.20 and that by that same Lemma $b \Rightarrow b \wedge (b_1 \vee b_2)$ and thus $b \Rightarrow b_1 \vee b_2$.

$$\begin{array}{cl} & p \\ = & \overrightarrow{\sum} b \rightarrow \alpha \\ \doteq & \overrightarrow{\sum} b \wedge (b_1 \vee b_2) \rightarrow \alpha \\ \doteq & \overrightarrow{\sum} (b \wedge b_1) \vee (b \wedge b_2) \rightarrow \alpha \\ \doteq & \overrightarrow{\sum} b \wedge b_1 \rightarrow \alpha + \overrightarrow{\sum} b \wedge b_2 \rightarrow \alpha \\ \doteq & p_1 + p_2 \end{array}$$

- $p = \overrightarrow{\sum} b \rightarrow \alpha \cdot r$, which is similar to the previous case
- $p = r + r'$

$$
\begin{array}{ll}
 & p \,\underline{\rightharpoonup}\, q+q' \\
\equiv & r+r' \,\underline{\rightharpoonup}\, q+q' \\
\equiv & r \,\underline{\rightharpoonup}\, q+q' \wedge r' \,\underline{\rightharpoonup}\, q+q' \\
\equiv & \exists_{s,t}(r \doteq s+t \wedge \#(s) \leq \#(r) \wedge \#(t) \leq \#(r) \wedge s \,\underline{\rightharpoonup}\, q \wedge t \,\underline{\rightharpoonup}\, q') \wedge \\
 & \exists_{s',t'}(r' \doteq s'+t' \wedge \#(s') \leq \#(r') \wedge \#(t') \leq \#(r') \wedge s' \,\underline{\rightharpoonup}\, q \wedge t' \,\underline{\rightharpoonup}\, q') \\
\equiv & \exists_{s,s',t,t'}(r \doteq s+t \wedge \#(s) \leq \#(r) \wedge \#(t) \leq \#(r) \wedge s \,\underline{\rightharpoonup}\, q \wedge t \,\underline{\rightharpoonup}\, q' \wedge \\
 & \quad r' \doteq s'+t' \wedge \#(s') \leq \#(r') \wedge \#(t') \leq \#(r') \wedge s' \,\underline{\rightharpoonup}\, q \wedge t' \,\underline{\rightharpoonup}\, q') \\
\Rightarrow & \exists_{s,s',t,t'}(r+r' \doteq s+t+s'+t' \wedge \#(s+s') \leq \#(r+r') \wedge \#(t+t') \leq \#(r+r') \wedge \\
 & \quad s+s' \,\underline{\rightharpoonup}\, q \wedge t+t' \,\underline{\rightharpoonup}\, q') \\
\Rightarrow & \exists_{s,t}(p \doteq s+t \wedge \#(s) \leq \#(p) \wedge \#(t) \leq \#(p) \wedge s \,\underline{\rightharpoonup}\, q \wedge t \,\underline{\rightharpoonup}\, q')
\end{array}
$$

□

The following final lemmas describe some additional facts needed for the completeness proof.

Lemma 7.23. Let b and c be boolean expressions and $\vec{d}$ and $\vec{e}$ vectors of data variables. Also, let $\alpha, \beta \in \mathbb{A}$. The following holds:

$$\textstyle\sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \,\underline{\rightharpoonup}\, \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e}) \Rightarrow \forall_{\vec{d}}(b(\vec{d}) \Rightarrow \exists_{\vec{e}}(c(\vec{e}) \wedge [\![\alpha(\vec{d})]\!] = [\![\beta(\vec{e})]\!])$$

Proof 7.23. This follows from the definition of $\underline{\rightharpoonup}$. □

Lemma 7.24. Let b and c be boolean expressions and $\vec{d}$ and $\vec{e}$ vectors of data variables. Also, let $\alpha, \beta \in \mathbb{A}$ and $p, q \in T_{pc}$. The following holds:

$$\textstyle\sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \cdot p(\vec{d}) \,\underline{\rightharpoonup}\, \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot q(\vec{e}) \Rightarrow \forall_{\vec{d}}(b(\vec{d}) \Rightarrow \exists_{\vec{e}}(c(\vec{e}) \wedge [\![\alpha(\vec{d})]\!] =$$
$$[\![\beta(\vec{e})]\!] \wedge p(\vec{d}) \underline{\leftrightarrow} q(\vec{e}))$$

Proof 7.24. This follows from the definition of $\underline{\rightharpoonup}$. □

Lemma 7.25. Let b and c be boolean expressions and $p \in T_{pc}$. The following holds:

$$\overrightarrow{\textstyle\sum} b \wedge c \rightarrow p \,\dot{\leqslant}\, \overrightarrow{\textstyle\sum} b \rightarrow p$$

Proof 7.25.

$$
\begin{array}{ll}
 & \overrightarrow{\sum} b \wedge c \rightarrow p + \overrightarrow{\sum} b \rightarrow p \\
\doteq & \overrightarrow{\sum} (b \wedge c) \vee b \rightarrow p \\
\doteq & \overrightarrow{\sum} (b \wedge c) \vee (b \wedge \mathbf{t}\,) \rightarrow p \\
\doteq & \overrightarrow{\sum} b \wedge (c \vee \mathbf{t}\,) \rightarrow p \\
\doteq & \overrightarrow{\sum} b \wedge \mathbf{t} \rightarrow p \\
\doteq & \overrightarrow{\sum} b \rightarrow p
\end{array}
$$

□

Theorem 7.26. Let $p, q \in T_{pc}$ and $\mathcal{A}$ be complete and have equality and quantifier elimination. GenSpect is (relatively) complete (i.e. $G \vDash p \underline{\leftrightarrow} q \Rightarrow G \vdash p \doteq q$).

Proof 7.26. Given $p \underline{\leftrightarrow} q$, we need to show that $p \doteq q$. As there are basic terms p' and q' with $p \doteq p' \wedge q \doteq q'$ (and, because $\doteq$ is sound, $p \underline{\leftrightarrow} p' \wedge q \underline{\leftrightarrow} q'$), it is sufficient to show that $p' \doteq q'$. We do this by proving that $p' \,\underline{\rightharpoonup}\, q' \Rightarrow p' \,\dot{\leqslant}\, q'$ and $q' \,\underline{\rightharpoonup}\, p' \Rightarrow q' \,\dot{\leqslant}\, p'$. From this obviously follows $p \underline{\leftrightarrow} q \equiv p' \underline{\leftrightarrow} q' \equiv p' \,\underline{\rightharpoonup}\, q' \wedge q' \,\underline{\rightharpoonup}\, p' \Rightarrow p' \,\dot{\leqslant}\, q' \wedge q' \,\dot{\leqslant}\, p' \equiv p' \doteq q'$.

We now prove $p' \,\underline{\rightharpoonup}\, q' \Rightarrow p' \,\dot{\leqslant}\, q'$ with induction on the number of symbols in p' and q' (with $\sum_{\vec{d}}$ and $\sum_{\vec{e}}$ a sequence of summations binding the variables $\vec{d}$ resp. $\vec{e}$, $b(\vec{d})$ and $c(\vec{e})$ boolean expressions depending on $\vec{d}$ resp. $\vec{e}$, and $\alpha(\vec{d})$ and $\beta(\vec{e})$ multiactions depending on $\vec{d}$ resp. $\vec{e}$):

- $p' = \sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d})$, which means that by induction on q'
  - $q' = \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e})$, which means that $\forall_{\vec{d}}(b(\vec{d}) \Rightarrow \exists_{\vec{e}}(c(\vec{e})) \wedge \forall_{\vec{e}}(c(\vec{e}) \Rightarrow [\![\alpha(\vec{d})]\!] = [\![\beta(\vec{e})]\!]))$ because of $p' \,\underline{\rightarrow}\, q'$ (by Lemma 7.23) and therefore $p' = \sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \wedge c(\vec{e}) \rightarrow \alpha(\vec{d}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \wedge c(\vec{e}) \rightarrow \beta(\vec{e}) \mathrel{\dot{\leqslant}} \sum_{\vec{e}} \sum_{\vec{d}} c(\vec{e}) \rightarrow \beta(\vec{e}) \doteq \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e}) \doteq q'$, or
  - $q' = \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot r(\vec{e})$, with basic term r, which means that $\forall_{\vec{d}}(\neg b(\vec{d}))$ (as q' cannot terminate) and $p' \doteq \delta \mathrel{\dot{\leqslant}} q'$, or
  - $q' = r + r'$, with basic terms r and r', which means that by Lemma 7.22 $\exists_{s,s'}(p' \doteq s + s' \wedge \#(s) \leq \#(p') \wedge \#(s') \leq \#(p') \wedge s \,\underline{\rightarrow}\, r \wedge s' \,\underline{\rightarrow}\, r')$ and by induction $\exists_{s,s'}(p' \doteq s + s' \wedge s \mathrel{\dot{\leqslant}} r \wedge s' \mathrel{\dot{\leqslant}} r') \Rightarrow \exists_{s,s'}(p' \doteq s + s' \wedge s \mathrel{\dot{\leqslant}} r + r' \wedge s' \mathrel{\dot{\leqslant}} r + r') \Rightarrow \exists_{s,s'}(p' \doteq s + s' \wedge s + s' \mathrel{\dot{\leqslant}} r + r') \Rightarrow p' \mathrel{\dot{\leqslant}} r + r' = q'$.
- $p' = \sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \cdot r(\vec{d})$, with basic term r, which means that by induction on q'
  - $q' = \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e})$, which means that $\forall_{\vec{d}}(\neg b(\vec{d}))$ (as p' cannot terminate) and $p' \doteq \delta \mathrel{\dot{\leqslant}} q'$, or
  - $q' = \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot s(\vec{e})$, which means that $\forall_{\vec{d}}(b(\vec{d}) \Rightarrow \exists_{\vec{e}}(c(\vec{e})) \wedge \forall_{\vec{e}}(c(\vec{e}) \Rightarrow [\![\alpha(\vec{d})]\!] = [\![\beta(\vec{e})]\!] \wedge r(\vec{d}) \,\underline{\leftrightarrow}\, s(\vec{e}))$ because of $p' \,\underline{\rightarrow}\, q'$ (by Lemmas 7.24 and 7.23) and by induction we have $r(\vec{d}) \,\underline{\rightarrow}\, s(\vec{e}) \wedge s(\vec{d}) \,\underline{\rightarrow}\, r(\vec{e}) \Rightarrow r(\vec{d}) \mathrel{\dot{\leqslant}} s(\vec{e}) \wedge s(\vec{e}) \mathrel{\dot{\leqslant}} r(\vec{d})$, and therefore $p' = \sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \cdot r(\vec{d}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \rightarrow \alpha(\vec{d}) \cdot r(\vec{d}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \wedge c(\vec{e}) \rightarrow \alpha(\vec{d}) \cdot r(\vec{d}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \wedge c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot r(\vec{e}) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \wedge c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot (r(\vec{e}) + s(\vec{e})) \doteq \sum_{\vec{e}} \sum_{\vec{d}} b(\vec{d}) \wedge c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot s(\vec{e}) \mathrel{\dot{\leqslant}} \sum_{\vec{e}} \sum_{\vec{d}} c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot s(\vec{e}) \doteq \sum_{\vec{e}} c(\vec{e}) \rightarrow \beta(\vec{e}) \cdot s(\vec{e}) \doteq q'$, or
  - $q' = s + s'$, with basic terms s and s', which means that by Lemma 7.22 $\exists_{t,t'}(p' \doteq t + t' \wedge \#(t) \leq \#(p') \wedge \#(t') \leq \#(p') \wedge t \,\underline{\rightarrow}\, s \wedge t' \,\underline{\rightarrow}\, s')$ and by induction $\exists_{t,t'}(p' \doteq t + t' \wedge t \mathrel{\dot{\leqslant}} s \wedge t' \mathrel{\dot{\leqslant}} s') \Rightarrow \exists_{t,t'}(p' \doteq t + t' \wedge t \mathrel{\dot{\leqslant}} s + s' \wedge t' \mathrel{\dot{\leqslant}} s + s') \Rightarrow \exists_{t,t'}(p' \doteq t + t' \wedge t + t' \mathrel{\dot{\leqslant}} s + s') \Rightarrow p' \mathrel{\dot{\leqslant}} s + s' = q'$.
- $p' = r + r'$, with basic terms r and r', which means that $p' \,\underline{\rightarrow}\, q' \equiv r + r' \,\underline{\rightarrow}\, q' \equiv r \,\underline{\rightarrow}\, q' \wedge r' \,\underline{\rightarrow}\, q' \overset{IH}{\equiv} r \mathrel{\dot{\leqslant}} q' \wedge r' \mathrel{\dot{\leqslant}} q' \equiv r + r' \mathrel{\dot{\leqslant}} q' \equiv p' \mathrel{\dot{\leqslant}} q'$.

The proof of $q' \,\underline{\rightarrow}\, p' \Rightarrow q' \mathrel{\dot{\leqslant}} p'$ is symmetrical to the proof above. □

8 Abstraction

If we want τ (or $\langle\rangle$) to be the "real" *silent step*, we want to be able to remove τ where its presence can not be determined. Our current definition of bisimulation therefore no longer suits us and we therefore introduce another form of bisimulation.

We use rooted branching bisimulation $\underline{\leftrightarrow}_{rb}$ and, as our semantics fits the "RBB cool" format[3], we know $\underline{\leftrightarrow}_{rb}$ is a congruence. To be able to give a nice definition we introduce a termination predicate $\downarrow$ and assume that $\checkmark$ is a state (but not an allowed process constant) and that it is the only state for which $\downarrow$ holds (i.e. $x \xrightarrow{m} \checkmark$ now just means that x can make a transition m to $\checkmark$ and $\checkmark$ terminates).

Definition 8.1. Let $G = \mathit{Sem}(\mathcal{A}, E) = \langle T_P, \mathbb{B}(\mathcal{A}), \longrightarrow, \longrightarrow \checkmark\rangle$, with $\mathcal{A}$ a data algebra, E a set of process expressions and $\longrightarrow$ and $\longrightarrow \checkmark$ the transition relations on processes T_P with multiactions $\mathbb{B}(\mathcal{A})$, be a process semantics. Also, let t,t',u and u' be process terms in which process variables may occur only if they are in E and $m \in \mathbb{B}(A)$. *Branching bisimulation* $\underline{\leftrightarrow}_b$ is the union of all relations B such that if tBu (with $\xrightarrow{m^*}$ zero or more m-transitions):

- for all t' and m, $t \xrightarrow{m} t'$ means that $m = [] \wedge t'Bu$ or there exist u' and u'' with $u \xrightarrow{[]^*} u'' \xrightarrow{m} u' \wedge tBu'' \wedge t'Bu'$
- for all u' and m, $u \xrightarrow{m} u'$ means that $m = [] \wedge tBu'$ or there exist t' and t'' with $t \xrightarrow{[]^*} t'' \xrightarrow{m} t' \wedge t''Bu \wedge t'Bu'$
- $t \downarrow$ means that there exists a u' with $u \xrightarrow{[]^*} u' \downarrow \wedge tBu'$
- $u \downarrow$ means that there exists a u' with $t \xrightarrow{[]^*} t' \downarrow \wedge t'Bu$

Definition 8.2. Let $G = \mathit{Sem}(\mathcal{A}, E) = \langle T_P, \mathbb{B}(A), \longrightarrow, \longrightarrow \surd \rangle$, with $\mathcal{A}$ a data algebra, E a set of process expressions and $\longrightarrow$ and $\longrightarrow \surd$ the transition relations on processes T_P with multiactions $\mathbb{B}(A)$, be a process semantics. Also, let t,t',u and u' be process terms in which process variables may occur only if they are in E and $m \in \mathbb{B}(A)$. *Rooted branching bisimulation* $\underline{\leftrightarrow}_{rb}$ is defined by $t \underline{\leftrightarrow}_{rb} u$ if, and only if, $t \underline{\leftrightarrow}_b u \wedge \mathit{rooted}(t, u)$, with $\mathit{rooted}(t, u)$ defined as follows:

- if $t \xrightarrow{m} t'$, then $u \xrightarrow{m} u' \wedge t' \underline{\leftrightarrow}_b u'$
- if $u \xrightarrow{m} u'$, then $t \xrightarrow{m} t' \wedge t' \underline{\leftrightarrow}_b u'$
- if $t \downarrow$, then $u \downarrow$
- if $u \downarrow$, then $t \downarrow$

Now we have this new form of equivalence, we also need a matching (that is sound and complete) axiomatisation. Fortunately, the axioms given before are still sound, but to make the axiomatisation (relatively) complete again (i.e. to have axioms that reflect the behaviour of τ) we believe it is sufficient to add the following two axioms, as in [4]. (Note that x, y and z cannot be $\surd$).

$T1 \quad x \cdot \tau = x$
$T2 \quad x \cdot (\tau \cdot (y + z) + y) = x \cdot (y + z)$

In Appendix B we have proved the soundness of the new axioms $T1$ and $T2$. The other axioms of G, which have already been proven to be sound with respect to $\underline{\leftrightarrow}$, do not need to be proven again, as $\underline{\leftrightarrow} \subset \underline{\leftrightarrow}_{rb}$ holds. The structure of the proofs of $T1$ and $T2$ is the same as before, but instead of proving R to be a bisimulation we have proved R to be a branching bisimulation and $\mathit{rooted}(t, u)$ (for axiom $t = u$).

When working with abstraction, one might encounter processes of the following form: $X = i \cdot X + Y$, where one wishes to hide i. This process can in fact do an infinite amount of i's, but, after hiding, these actions should not be observable. One usually wishes to have some form of fairness in which a infinite sequence of τ's is not possible. To express this we introduce a fairness rule.

In current algebras, like ACP, there exists rules as "KFAR" and "CFAR"[2]. Although these rules basically express the same fairness as we wish to express and fit in our own language, it needs that the (recursive) specification is linearised (before the application of the abstraction operator τ_I).

Instead we wish to see if it is possible to first apply the abstraction and afterwards apply some form of fairness rule to the (possibly, or even probably) reduced specification. The proposition of the basic idea follows. It is clear that if this rule is suitable, we can extend it to a more general rule (as "CFAR").

Proposition 8.3. Let Y be a guarded recursive specification. The *fairness rule* we propose is the following:

$$\frac{Y \doteq \tau_{\{i\}}(Y), X' \doteq i \cdot X' + Y, X \doteq \tau_{\{i\}}(X')}{\tau \cdot X \doteq \tau \cdot Y}$$

This rule basically follows from KFAR, as shown in the following derivation:

$$\frac{\dfrac{Y \doteq \tau_{\{i\}}(Y), X' \doteq i \cdot X' + Y, X \doteq \tau_{\{i\}}(X')}{Y \doteq \tau_{\{i\}}(Y), \tau \cdot \tau_{\{i\}}(X') \doteq \tau \cdot \tau_{\{i\}}(Y), X \doteq \tau_{\{i\}}(X')}}{\tau \cdot X \doteq \tau \cdot Y}$$

One might wonder why the X is defined as an abstraction of X' and not just as $X = \tau \cdot X + Y$. The problem with this equation is that, for example, $X = \tau \cdot a$, $Y = \delta$ is a solution, but with the abstraction of X' this is not the case. This does mean, however, that our wish to be able to simply apply a fair abstraction rule to a abstracted specification is not possible, unless we know that the variable in question satisfies the conditions of the above rule. Fortunately, this will be the case in practice, as the abstracted specification is an abstraction of some specification. It should therefore be save to use this rule as the following, if one knows the occurring τ is a result of abstraction:

$$\frac{X = \tau X + Y}{\tau X = \tau Y}$$

Another method, that is also usable with more complex processes (with data), is the *cones and foci* method[6]. This method takes a specification (without silent steps), an implementation (in practice usually with silent steps) and a mapping from states in the implementation to states in the specification, and states that these are bisimilar for all states that satisfy a certain invariant, if that invariant implies the following *matching criteria*:

a. The implementation is convergent (i.e. there is a well-founded order on the state, such that after each silent step the state is smaller than the previous state).

b. If a silent step can occur in the implementation, the mapped state does not change.

c. If a non-silent step can occur in the implementation, than it should also be able to occur in the specification (i.e. with the same arguments and with the same resulting state).

d. If no silent step can occur in the implementation and a step can occur in the specification, than that same step should be able to occur in the implementation.

For the precise details see [6]. In the examples we use the steps as indicated in [6] (step 1 corresponds to a, 2 to b, 3, 5 and 6 to c and 4 to d).

9 Example: Alternating Bit Protocol

In the previous sections we have given a formal definition of a new algebra and proven its axiomatisation sound and (relatively) complete. We now have a look at some examples in which we use this new algebra. The first example is the Alternating Bit Protocol (as in [2]).

Assume we have two processes S and R, that need to communicate over unreliable links K and L. Process S receives a sequence of data with action r_1 and the goal is that process R will send that same data, in the same order as it was received by S, with action s_4 (i.e. the protocol should make the system of S, R, K and L behave like a buffer). When the data of the communication over a link gets

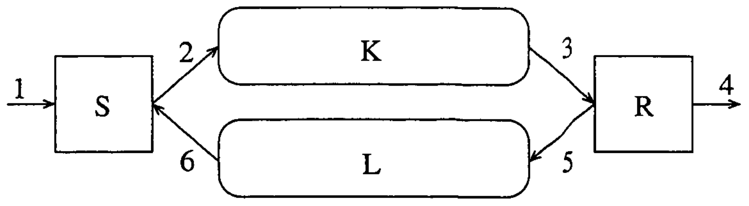

Figure 1: Alternating Bit Protocol.

corrupted or gets lost, the receiving party will be able to detect this. (These cases are represented by $\perp$.)
The Alternating Bit Protocol establishes this goal by adding a bit to the data before transmission by S and having R send back that bit indicating the success of the transmission, or the inverse of the bit in case of failure. The moment S receives the correct bit, it knows it can send the next datum, or, in the case it receives the other bit, it sends the current datum another time.

We assume the existence of some data type D, and a type Bit consisting of 0, 1 and $\perp$, for which we use variable n. Also we assume a data type F that combines these two types (i.e. F contains tuples of a datum and a bit).

We define a one place buffer B, receiving data with action r_1 and sending data with s_4 as follows:

$$B = \sum_{d:D} r_1(d) \cdot s_4(d) \cdot B$$

Next is the definition of the processes of the Alternating Bit Protocol:

$$\begin{array}{lcl}
S & = & S(0) \cdot S(1) \cdot S \\
S(n{:}Bit) & = & \sum_{d:D} r_1(d) \cdot S(n,d) \\
S(n{:}Bit, d{:}D) & = & s_2(dn) \cdot T(n,d) \\
T(n{:}Bit, d{:}D) & = & (r_6(1-n) + r_6(\perp)) \cdot S(n,d) + r_6(n) \\
 & & \\
R & = & R(1) \cdot R(0) \cdot R \\
R(n{:}Bit) & = & (\sum_{d:D} r_3(dn) + r_3(\perp)) \cdot s_5(n) \cdot R(n) + \sum_{d:D} r_3(d(1-n)) \cdot s_4(d) \cdot s_5(1-n) \\
 & & \\
K & = & \sum_{x:F} r_2(x)(i \cdot s_3(x) + i \cdot s_3(\perp)) \cdot K \\
 & & \\
L & = & \sum_{n:\{0,1\}} r_5(n)(i \cdot s_6(n) + i \cdot s_6(\perp)) \cdot L
\end{array}$$

To get the whole system (A), we put processes S, K, L and R in parallel and force communications on channels 2,3,5 and 6. By hiding all internal actions (i.e. every action but those on channels 1 and 4) we get A_τ.

$$\begin{array}{lcl}
A & = & \nabla_V(\Gamma_C(S \parallel K \parallel L \parallel R)) \\
C & = & \{s_2|r_2 \rightarrow c_2, s_3|r_3 \rightarrow c_3, s_5|r_5 \rightarrow c_5, s_6|r_6 \rightarrow c_6\} \\
V & = & \{i, r_1, s_4, c_2, c_3, c_5, c_6\} \\
 & & \\
A_\tau & = & \tau_I(A) \\
I & = & \{i, c_2, c_3, c_5, c_6\}
\end{array}$$

We eliminate the complex operators ∇, Γ and $\parallel$ from A using a standard expansion and introducing $A(d:D)$ and $A_2(d:D)$ as we go along.

$$\begin{array}{lcl}
A & = & \nabla_V(\Gamma_C(S \parallel K \parallel L \parallel R)) \\
 & \doteq & \sum_{d:D} r_1(d) \cdot \nabla_V(\Gamma_C(S(0,d) \cdot S(1) \cdot S \parallel K \parallel L \parallel R)) \\
 & \doteq & \sum_{d:D} r_1(d) \cdot A(d) \\
 & & \\
A(d{:}D) & = & \nabla_V(\Gamma_C(S(0,d) \cdot S(1) \cdot S \parallel K \parallel L \parallel R)) \\
 & \doteq & c_2(d0) \cdot \nabla_V(\Gamma_C(T(0,d) \cdot S_1 \cdot S \parallel (i \cdot s_3(d0) + i \cdot s_3(\perp)) \cdot K \parallel L \parallel R))
\end{array}$$

$$\begin{array}{lcl}
 & \doteq & c_2(d0)\cdot(i\cdot c_3(d0)\cdot s_4(d)\cdot\nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel s_5(0)\cdot R(0)\cdot R))+ \\
 & & i\cdot c_3(\perp)\cdot c_5(1)\cdot\nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel K \parallel (i\cdot s_6(1)+i\cdot s_6(\perp))\cdot L \parallel R))) \\
 & \doteq & c_2(d0)\cdot(i\cdot c_3(d0)\cdot s_4(d)\cdot A_2(d)+ \\
 & & i\cdot c_3(\perp)\cdot c_5(1)\cdot(i\cdot c_6(1)\cdot\nabla_V(\Gamma_C(S(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel R))+ \\
 & & i\cdot c_6(\perp)\cdot\nabla_V(\Gamma_C(S(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel R)))) \\
 & \doteq & c_2(d0)\cdot(i\cdot c_3(d0)\cdot s_4(d)\cdot A_2(d)+i\cdot c_3(\perp)\cdot c_5(1)\cdot(i\cdot c_6(1)+i\cdot c_6(\perp))\cdot A(d))
\end{array}$$

$$\begin{array}{lcl}
A_2(d{:}D) & = & \nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel s_5(0)\cdot R(0)\cdot R)) \\
 & \doteq & c_5(0)\cdot\nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel K \parallel (i\cdot s_6(0)+i\cdot s_6(\perp))\cdot L \parallel R(0)\cdot R)) \\
 & \doteq & c_5(0)\cdot(i\cdot c_6(0)\cdot\nabla_V(\Gamma_C(S(1)\cdot S \parallel K \parallel L \parallel R(0)\cdot R))+ \\
 & & i\cdot c_6(\perp)\cdot\nabla_V(\Gamma_C(S(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel R(0)\cdot R))) \\
 & \doteq & c_5(0)\cdot(i\cdot c_6(0)\cdot A'+ \\
 & & i\cdot c_6(\perp)\cdot c_2(d0)\cdot\nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel (i\cdot s_3(d0)+i\cdot s_3(\perp))\cdot K \parallel L \parallel R(0)\cdot R))) \\
 & \doteq & c_5(0)\cdot(i\cdot c_6(0)\cdot A'+ \\
 & & i\cdot c_6(\perp)\cdot c_2(d0)\cdot(i\cdot c_3(d0)\cdot\nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel s_5(0)\cdot R(0)\cdot R))+ \\
 & & i\cdot c_3(\perp)\cdot\nabla_V(\Gamma_C(T(0,d)\cdot S(1)\cdot S \parallel K \parallel L \parallel s_5(0)\cdot R(0)\cdot R)))) \\
 & \doteq & c_5(0)\cdot(i\cdot c_6(0)\cdot A'+i\cdot c_6(\perp)\cdot c_2(d0)\cdot(i\cdot c_3(d0)+i\cdot c_3(\perp))\cdot A_2(d))
\end{array}$$

The same calculation can be made for A', which will result in the following:

$$\begin{array}{lcl}
A' & \doteq & \sum_{d:D} r_1(d)\cdot A'(d) \\
A'(d{:}D) & \doteq & c_2(d1)\cdot(i\cdot c_3(d1)\cdot s_4(d)\cdot A'_2(d)+i\cdot c_3(\perp)\cdot c_5(0)\cdot(i\cdot c_6(0)+i\cdot c_6(1))\cdot A'(d)) \\
A'_2(d{:}D) & \doteq & c_5(1)\cdot(i\cdot c_6(1)\cdot A+i\cdot c_6(\perp)\cdot c_2(d1)\cdot(i\cdot c_3(d1)+i\cdot c_3(\perp))\cdot A'_2(d))
\end{array}$$

Now we apply the abstraction operator to obtain A_τ (and $A_\tau(d:D)$ and $A_{\tau 2}(d:D)$).

$$\begin{array}{lcl}
A_\tau & = & \tau_I(A) \\
 & \doteq & \sum_{d:D} r_1(d)\cdot\tau_I(A_d) \\
 & \doteq & \sum_{d:D} r_1(d)\cdot A_\tau(d)
\end{array}$$

$$\begin{array}{lcl}
A_\tau(d{:}D) & = & \tau_I(A_d) \\
 & \doteq & \tau\cdot(\tau\cdot\tau\cdot s_4(d)\cdot\tau_I(A_2(d))+\tau\cdot\tau\cdot\tau\cdot(\tau\cdot\tau+\tau\cdot\tau)\cdot\tau_I(A(d))) \\
 & \doteq & \tau\cdot(\tau\cdot s_4(d)\cdot A_{\tau 2}(d)+\tau\cdot A_\tau(d))
\end{array}$$

$$\begin{array}{lcl}
A_{\tau 2}(d{:}D) & = & \tau_I(A_2(d)) \\
 & \doteq & \tau\cdot(\tau\cdot\tau\cdot\tau_I(A')+\tau\cdot\tau\cdot\tau\cdot(\tau\cdot\tau+\tau\cdot\tau)\cdot\tau_I(A_2(d))) \\
 & \doteq & \tau\cdot(\tau\cdot A'_\tau+\tau\cdot A_{\tau 2}(d))
\end{array}$$

And again the same for A'_τ.

$$\begin{array}{lcl}
A'_\tau & \doteq & \sum_{d:D} r_1(d)\cdot A'_\tau(d) \\
A'_\tau(d{:}D) & \doteq & \tau\cdot(\tau\cdot s_4(d)\cdot A'_{\tau 2}(d)+\tau\cdot A'_\tau(d)) \\
A'_{\tau 2}(d{:}D) & \doteq & \tau\cdot(\tau\cdot A_\tau+\tau\cdot A'_{\tau 2}(d))
\end{array}$$

As one could (or should) have expected, it is clear that this process can do an infinite sequence of τ's, which corresponds to the transmissions over a link continuously failing. In practice we usually assume, or know, that this will actually never happen. With the proposed fairness rule from the previous section we can say the following (implicitly introducing X and Y):

$$\begin{array}{lcl}
A_\tau(d{:}D) & = & \tau\cdot(\tau\cdot s_4(d)\cdot A_{\tau 2}(d)+\tau\cdot A_\tau(d)) \\
 & \doteq & \tau\cdot(X+\tau\cdot Y)
\end{array}$$

$$\begin{array}{lcl} & \doteq & \tau \cdot Y \\ \\ X & = & \tau \cdot s_4(d) \cdot A_{\tau 2}(d) \\ \\ Y & = & (X + \tau \cdot Y) \\ \tau Y & \doteq & \tau X \\ \\ A_\tau(d{:}D) & \doteq & \tau \cdot X \\ & \doteq & \tau \cdot s_4(d) \cdot A_{\tau 2}(d) \end{array}$$

And the same for $A_{\tau 2}(d{:}D)$,$A'_\tau(d{:}D)$ and $A'_{\tau 2}(d{:}D)$. This gives us the following specification:

$$\begin{array}{lcl} A_\tau & = & \sum_{d:D} r_1(d) \cdot A_\tau(d) \\ A_\tau(d{:}D) & = & \tau \cdot s_4(d) \cdot A_{\tau 2}(d) \\ A_{\tau 2}(d{:}D) & = & \tau \cdot A'_\tau \\ A'_\tau & = & \sum_{d:D} r_1(d) \cdot A'_\tau(d) \\ A'_\tau(d{:}D) & = & \tau \cdot s_4(d) \cdot A'_{\tau 2}(d) \\ A'_{\tau 2}(d{:}D) & = & \tau \cdot A_\tau \end{array}$$

With substitution and RSP we can conclude our calculations:

$$\begin{array}{lcl} A_\tau & = & \sum_{d:D} r_1(d) \cdot A_\tau(d) \\ & = & \sum_{d:D} r_1(d) \cdot \tau \cdot s_4(d) \cdot A_{\tau 2}(d) \\ & = & \sum_{d:D} r_1(d) \cdot s_4(d) \cdot A'_\tau \\ A'_\tau & = & \sum_{d:D} r_1(d) \cdot A'_\tau(d) \\ & = & \sum_{d:D} r_1(d) \cdot \tau \cdot s_4(d) \cdot A'_{\tau 2}(d) \\ & = & \sum_{d:D} r_1(d) \cdot s_4(d) \cdot A_\tau \\ \\ A_\tau & = & \sum_{d:D} r_1(d) \cdot s_4(d) \cdot A'_\tau \\ & = & \sum_{d:D} r_1(d) \cdot s_4(d) \cdot A_\tau \end{array}$$

As we can see, the (abstracted) Alternating Bit Protocol is, under the assumption of the fairness rule, just the one place buffer B (by RSP).

10 Examples: Petri-nets

The following examples show how we can translate Petri-nets to GenSpect and calculate with the result. We believe that any Petri-net is easily translatable by writing processors as an alternative composition of multiactions, based on the input/output relations they describe. Places would be translated into an alternative composition of all possible combinations of taking and putting tokens that can occur simultaneously.

10.1 Squares

Assume we have a processor which can take a token with some value n from one place (connected with a channel i) and (simultaneously) puts a token with value n^2 in a(nother) place (connected with a channel j). We can describe this processor in the following way:

$$Sqr_{ij} = \sum_{n:\mathbb{N}} \overline{get}_i(n) | \overline{put}_j(n^2) \cdot Sqr_{ij}$$

To calculate n^4 we can now connect two instances of Sqr with a place (Figure 2).

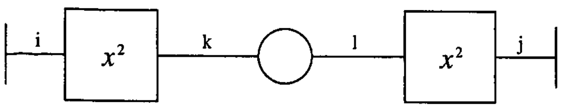

Figure 2: Square functions connected by a place.

First we define what a place is. Places are basically bags, from or to which connected processors can take or give values. We describe a place, with an (incoming) channel i and (outgoing) channel j, as follows (assuming a type Bag of bags over $\mathbb{N}$, and operators $\oplus, \ominus, \circledcirc$, which add an element to, remove an element from or check whether or not an element occurs in a bag respectively; note that these types and operators are just introduced for this example):

$$P_{ij}(b : Bag) = \sum_{n:\mathbb{N}} \underline{put}_i(n) \cdot P_{ij}(n \oplus b) + \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \underline{get}_j(n) \cdot P_{ij}(n \ominus b)$$

Note that the definition of a place we use here does not allow tokens to be taken from and added to it simultaneously, because we want to keep the examples simple. The calculation of n^4 can now be described as follows (with $C = \{\overline{put}_k|\underline{put}_k \rightarrow put_k, \overline{get}_l|\underline{get}_l \rightarrow get_l\}$, $V = \{\overline{get}_i, put_k, get_l, \overline{put}_j\}$ and $[]$ the empty bag):

$$DSqr_{ij} = \nabla_V(\Gamma_C(Sqr_{i,k} \parallel P_{k,l}([]) \parallel Sqr_{l,j}))$$

If we now calculate with this equation, we get:

$$\begin{array}{lcl} DSqr_{ij} & = & DSqr'_{ij}([]) \\ DSqr'_{ij}(b : Bag) & = & \nabla_V(\Gamma_C(Sqr_{i,k} \parallel P_{k,l}(b) \parallel Sqr_{l,j})) \\ & \doteq & \sum_{n:\mathbb{N}} \langle \overline{get}_i(n), put_k(n^2) \rangle \cdot DSqr'_{i,j}(n^2 \oplus b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \langle get_l(n), \overline{put}_j(n^2) \rangle \cdot DSqr'_{ij}(n \ominus b) \end{array}$$

We see that this little system has a structure very similar to that of a place. Certainly if we would also hide the actions put_k and get_l ($I = \{put_k, get_l\}$):

$$\begin{array}{lcl} DSqr^{\tau}_{ij}(b : Bag) & = & \tau_I(DSqr'_{ij}(b)) \\ & \doteq & \sum_{n:\mathbb{N}} \langle \overline{get}_i(n) \rangle \cdot DSqr^{\tau}_{i,j}(n^2 \oplus b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \langle \overline{put}_j(n^2) \rangle \cdot DSqr^{\tau}_{ij}(n \ominus b) \\ & \doteq & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{i,j}(n^2 \oplus b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \overline{put}_j(n^2) \cdot DSqr^{\tau}_{ij}(n \ominus b) \end{array}$$

Note that in this specification no n^4 appears, because squares are put in the bag and are retrieved from of the bag as some number n. However, one can prove that this specification is equal to the following:

$$X_{ij}(b : Bag) = \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot X_{ij}(n \oplus b) + \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \overline{put}_j(n^4) \cdot X_{ij}(n \ominus b)$$

We prove that $X_{ij}(b) = DSqr^{\tau}_{ij}(b^2)$, where $[]^2 = []$ and $(n \oplus b)^2 = n^2 \oplus b^2$, by showing that $DSqr^{\tau}_{ij}(b^2)$ is a solution of $X_{jj}(b)$. (We use $\approx$ to denote a step that is justified by what is calculated later.)

$$\begin{array}{ll} & X_{ij}(b) \\ = & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot X_{ij}(n \oplus b) + \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \overline{put}_j(n^4) \cdot X_{ij}(n \ominus b) \\ \approx & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{ij}((n \oplus b)^2) + \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \overline{put}_j(n^4) \cdot DSqr^{\tau}_{ij}((n \ominus b)^2) \\ \doteq & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{ij}(n^2 \oplus b^2) + \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \overline{put}_j(n^4) \cdot DSqr^{\tau}_{ij}(n^2 \ominus b^2) \\ \doteq & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{ij}(n^2 \oplus b^2) + \sum_{m:\mathbb{N}} \sum_{n:\mathbb{N}} m = n^2 \wedge n^2 \circledcirc b^2 \rightarrow \overline{put}_j(n^4) \cdot DSqr^{\tau}_{ij}(n^2 \ominus b^2) \\ \doteq & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{ij}(n^2 \oplus b^2) + \sum_{m:\mathbb{N}} \sum_{n:\mathbb{N}} m = n^2 \wedge m \circledcirc b^2 \rightarrow \overline{put}_j(m^2) \cdot DSqr^{\tau}_{ij}(m \ominus b^2) \\ \doteq & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{ij}(n^2 \oplus b^2) + \sum_{m:\mathbb{N}} m \circledcirc b^2 \rightarrow \overline{put}_j(m^2) \cdot DSqr^{\tau}_{ij}(m \ominus b^2) \\ \doteq & \sum_{n:\mathbb{N}} \overline{get}_i(n) \cdot DSqr^{\tau}_{ij}(n^2 \oplus b^2) + \sum_{n:\mathbb{N}} n \circledcirc b^2 \rightarrow \overline{put}_j(m^2) \cdot DSqr^{\tau}_{ij}(n \ominus b^2) \\ = & DSqr^{\tau}_{ij}(b^2) \end{array}$$

10.2 Connected places

Assume we have two places. In standard petri-nets, we can only connect these with a processor. However, there is no such limitation in GenSpect. So the question rises: What happens if we connect two places together?

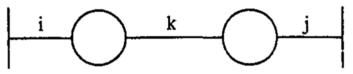

Figure 3: Two connected places.

Taking the definition of a place like before, we get (with $C = \{\underline{get}_k | \underline{put}_k \rightarrow pass_k\}$ and $V = \{\underline{put}_i, pass_k, \underline{get}_j\}$):

$$P^2 = \nabla_V(\Gamma_C(P_{ik}([]) \parallel P_{kj}([])))$$

Now, let us calculate:

$$\begin{array}{lcl} P^2 & = & P^{2'}([],[]) \\ P^{2'}(a,b:Bag) & = & \nabla_V(\Gamma_C(P_{ik}([]) \parallel P_{kj}([]))) \\ & \doteq & \sum_{n:\mathbb{N}} \underline{put}_i(n) \cdot P^{2'}(n \oplus a, b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc a \rightarrow pass_k(n) \cdot P^{2'}(n \ominus a, n \oplus b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \underline{get}_j(n) \cdot P^{2'}(a, n \ominus b) \end{array}$$

It is clear that we almost have the specification of a place again, the only difference being that that the original bag is now split up in two bags and the possibility to transfer elements from one bag to the other. After hiding $pass_k$, we can prove with the *cones and foci* method that, P and P^2 are in fact bisimilar.

$$\begin{array}{lcl} P^2_\tau & = & P^{2'}_\tau([],[]) \\ P^{2'}_\tau(a,b:Bag) & = & \tau_{\{pass_k\}}(P^{2'}(a,b)) \\ & \doteq & \sum_{n:\mathbb{N}} \underline{put}_i(n) \cdot P^{2'}(n \oplus a, b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc a \rightarrow \tau \cdot P^{2'}(n \ominus a, n \oplus b) + \\ & & \sum_{n:\mathbb{N}} n \circledcirc b \rightarrow \underline{get}_j(n) \cdot P^{2'}(a, n \ominus b) \end{array}$$

Let $P^{2'}_\tau(a,b)$ be the *implementation* and $P(a \cup b)$, with $n \circledcirc (a \cup b) \equiv n \circledcirc a \vee n \circledcirc b$, the *specification.*

1. $P^{2'}_\tau(a,b)$ is convergent; trivial with the smallest well-founded order that satisfies $\langle a, n \oplus b\rangle < \langle n \oplus a, b\rangle$.

2. $n \circledcirc a \Rightarrow (n \ominus a) \cup (n \oplus b)$, trivial (as $(n \ominus a) \cup (n \oplus b) = a \cup b$).

3. $n \circledcirc b \Rightarrow n \circledcirc (a \cup b)$, trivial.

4. $FC(\langle a,b\rangle) = \neg(n \circledcirc a)$; $FC(\langle a,b\rangle) \wedge n \circledcirc (a \cup b) \Rightarrow n \circledcirc b$, trivial.

5. Trivial.

6. $(n \oplus a) \cup b = n \oplus (a \cup b)$, trivial. $a \cup (n \ominus b) = n \ominus (a \cup b)$, trivial.

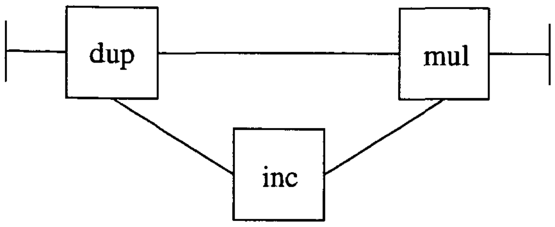

Figure 4: Three interconnected processors.

10.3 Connected processors

Assume we have three processors: *dup*, *inc* and *mul*. Processor *dup* receives numbers over one connector and sends the same number to the other two connectors, *inc* receives a number, increments it and then sends it, and *mul* receives two numbers and sends out the multiplication of those numbers. We connect these processors as shown in Figure 4 and are interested in the behaviour of the resulting system *dim*.
The definitions of these processors is as follows:

$$\begin{array}{lcl}
dup & = & \sum_{x:\mathbb{R}}(r_{dup}(x)|s_{dup_1}(x)|s_{dup_2}(x)) \cdot dup \\
inc & = & \sum_{x:\mathbb{R}}(r_{inc}(x)|s_{inc}(x+1)) \cdot inc \\
mul & = & \sum_{x:\mathbb{R}}\sum_{y:\mathbb{R}}(r_{mul_1}(x)|r_{mul_2}(y)|s_{mul}(x*y)) \cdot mul \\
C & = & \{s_{dup_1}|r_{mul_1}, s_{dup_2}|r_{inc}, s_{inc}|r_{mul_2}\} \\
V & = & \{r_{dup}|s_{mul}\} \\
dim & = & \nabla_V(\Gamma_C(dup \parallel inc \parallel mul))
\end{array}$$

We calculate with *dim*:

$$\begin{array}{lcl}
dim & = & \nabla_V(\Gamma_C(dup \parallel inc \parallel mul)) \\
 & \doteq & \nabla_V(\Gamma_C(dup \parallel inc \parallel mul)) \\
 & \doteq & \nabla_V(\Gamma_C(dup \mathbin{|\!\!\lfloor} (inc \parallel mul))) + \nabla_V(\Gamma_C((inc \parallel mul) \mathbin{|\!\!\lfloor} dup)) + \nabla_V(\Gamma_C(dup|(inc \parallel mul))) \\
 & \doteq & \delta + \nabla_V(\Gamma_C((inc \mathbin{|\!\!\lfloor} mul)) \mathbin{|\!\!\lfloor} dup) + \nabla_V(\Gamma_C((mul \mathbin{|\!\!\lfloor} inc) \mathbin{|\!\!\lfloor} dup)) + \\
 & & \nabla_V(\Gamma_C((inc \mathbin{|\!\!\lfloor} mul)) \mathbin{|\!\!\lfloor} dup) + \nabla_V(\Gamma_C(dup|(inc \mathbin{|\!\!\lfloor} mul))) + \nabla_V(\Gamma_C(dup|(mul \mathbin{|\!\!\lfloor} inc))) + \\
 & & \nabla_V(\Gamma_C(dup|(inc|mul))) \\
 & \doteq & \delta + \delta + \delta + \delta + \delta + \nabla_V(\Gamma_C(dup|inc|mul)) \\
 & \doteq & \nabla_V(\Gamma_C(\sum_{x:\mathbb{R}}\sum_{x':\mathbb{R}}\sum_{x'':\mathbb{R}}\sum_{y:\mathbb{R}}(\\
 & & \qquad (r_{dup}(x)|s_{dup_1}(x)|s_{dup_2}(x)) \cdot dup)| \\
 & & \qquad ((r_{inc}(x')|s_{inc}(x'+1)) \cdot inc)| \\
 & & \qquad ((r_{mul_1}(x'')|r_{mul_2}(y)|s_{mul}(x''*y)) \cdot mul))) \\
 & \doteq & \sum_{x:\mathbb{R}}\sum_{x':\mathbb{R}}\sum_{x'':\mathbb{R}}\sum_{y:\mathbb{R}} x = x' \wedge x = y \wedge x' + 1 = x'' \rightarrow (r_{dup}(x)|s_{mul}(x''*y)) \cdot \\
 & & \qquad \nabla_V(\Gamma_C(dup \parallel inc \parallel mul)) \\
 & \doteq & \sum_{x:\mathbb{R}}(r_{dup}(x)|s_{mul}((x+1)*x)) \cdot dim
\end{array}$$

As one can see, system *dim* has the same behaviour as a processor that takes an number x and returns $(x+1)*x$.

10.4 Distribution center

The system depicted in Figure 5 represents a distribution center, where orders enter. An order is represented as a pair $\langle i, n\rangle$, where i is a customer identification and n the number of ordered products. Orders are supposed to result in deliveries. A delivery is a pair $\langle i, n\rangle$ just like an order. The *ord* and

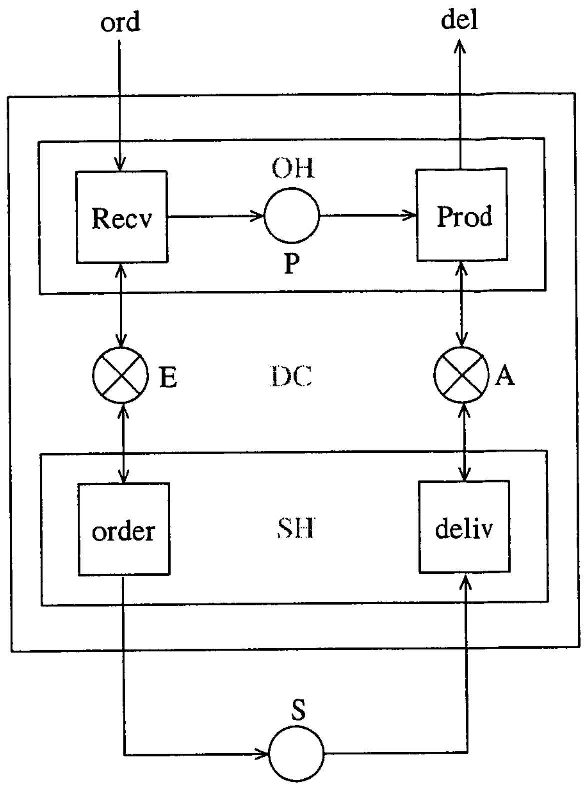

Figure 5: Distribution center.

deliv ports are connected to customers. The ports *supord* and *supdel* are connected to suppliers and represent respectively the ordering and delivery of a standard quantity σ $(0 < \sigma)$ of products.

The distribution center consists of two parts, order handling *OH* and stock handling *SH*. These parts interface via two stores representing the economic and actual stock. The actual and economic stock both contain initially α products. The order handler, upon receiving an order, subtracts the amount ordered from the economic stock and copies the order to an internal place. If the internal place contains an order with an amount less than the actual stock, the order is delivered and the amount ordered subtracted from the actual stock. The stock handler checks whether the economic stock is less than α $(0 < \alpha)$; if so, a supplier order is issued and the economic stock is incremented with σ products. If a supplier delivery comes in, the actual stock is incremented with σ products.

We want to analyse what the behaviour of the distribution center plus a reliable supplier is with respect to its customers. The reliable supplier $S(0)$ has the behaviour of a place: ordered items will always eventually be delivered. Its parameter is the number of pending orders.

The GenSpect model of the system *DCU* is $Q(\alpha, 0, \alpha, [])$ where $Q(n, k, m, X)$ for $n, k, m \in \mathbb{N}$ and $X \in \mathbb{B}(\mathcal{I} \times \mathbb{N})$ is defined by

$$
\begin{array}{lcl}
Q(n : \mathbb{Z}, k : \mathbb{N}, m : \mathbb{N}, X : \mathbb{B}(\mathcal{I} \times \mathbb{N})) & = & \nabla_{\{ord, del\}}(\Gamma_{\{supord|si \to \tau, supdel|so \to \tau\}}(DC(n, m, X) \parallel S(k))) \\
S(k : \mathbb{N}) & = & si \cdot S(k+1) + k > 0 \to so \cdot S(k-1) \\
DC(n : \mathbb{Z}, m : \mathbb{N}, X : \mathbb{B}(\mathcal{I} \times \mathbb{N})) & = & \nabla_{\{ord, del, supord, supdel\}}(\Gamma_{\{e|es \to \tau, a|as \to \tau\}}(\Gamma_{\{e|eo \to \tau, a|ao \to \tau\}}(\\
 & & \qquad OH(X) \parallel SH \parallel E(n) \parallel A(m)))) \\
E(n : \mathbb{Z}) & = & \sum_{m:\mathbb{N}} e(n, m) \cdot E(m) \\
A(n : \mathbb{N}) & = & \sum_{m:\mathbb{N}} a(n, m) \cdot A(m) \\
OH(X : \mathbb{B}(\mathcal{I} \times \mathbb{N})) & = & \nabla_{\{ord|eo, del|ao\}}(\Gamma_{\{p_1|p_2 \to \tau, p_3|p_4 \to \tau\}}(Recv \parallel P(X) \parallel Prod))
\end{array}
$$

$$\begin{array}{lcl}
P(X:\mathbb{B}(\mathcal{I}\times\mathbb{N})) & = & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}} p_2(i,s)\cdot P(\langle i,s\rangle \oplus X)+\\
 & & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\langle i,s\rangle \mathrel{\in\!\!\!\!\bigcirc} X \rightarrow p_3(i,s)\cdot P(\langle i,s\rangle \ominus X)\\
\mathit{Recv} & = & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}}\langle \mathit{ord}(i,s), \mathit{eo}(n,n-s), p_1(i,s)\rangle\cdot \mathit{Recv}\\
\mathit{Prod} & = & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s\rightarrow\langle \mathit{del}(i,s), \mathit{ao}(m,m-s), p_4(i,s)\rangle\cdot \mathit{Prod}\\
\mathit{SH} & = & \nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}\parallel \mathit{deliv})\\
\mathit{order} & = & \sum_{n:\mathbb{N}} n<\alpha\rightarrow\langle \mathit{es}(n,n+\sigma),\mathit{supord}\rangle\cdot \mathit{order}\\
\mathit{deliv} & = & \sum_{n:\mathbb{N}}\langle \mathit{as}(n,n+\sigma),\mathit{supdel}\rangle\cdot \mathit{deliv}
\end{array}$$

The first analysis step is to linearise the above specification.

$$\begin{array}{cl}
 & \mathit{SH}\\
= & \nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}\parallel \mathit{deliv})\\
\doteq & \nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}\parallel \mathit{deliv})\\
\doteq & \nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}\mathbin{|\!\lfloor} \mathit{deliv})+\\
 & \nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{deliv}\mathbin{|\!\lfloor} \mathit{order})+\\
 & \nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}|\mathit{deliv})\\
\doteq & \sum_{n:\mathbb{N}} n<\alpha\rightarrow\langle \mathit{es}(n,n+\sigma),\mathit{supord}\rangle\cdot\nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}\parallel \mathit{deliv})+\\
 & \sum_{n:\mathbb{N}}\langle \mathit{as}(n,n+\sigma),\mathit{supdel}\rangle\cdot\nabla_{\{\mathit{supord}|\mathit{es},\mathit{supdel}|\mathit{as}\}}(\mathit{order}\parallel \mathit{deliv})+\delta\\
\doteq & \sum_{n:\mathbb{N}} n<\alpha\rightarrow\langle \mathit{es}(n,n+\sigma),\mathit{supord}\rangle\cdot \mathit{SH}+\sum_{n:\mathbb{N}}\langle \mathit{as}(n,n+\sigma),\mathit{supdel}\rangle\cdot \mathit{SH}
\end{array}$$

$$\begin{array}{cl}
 & \mathit{OH}(X)\\
= & \nabla_{\{\mathit{ord}|\mathit{eo},\mathit{del}|\mathit{ao}\}}(\Gamma_{\{p_1|p_2\rightarrow\tau,p_3|p_4\rightarrow\tau\}}(\mathit{Recv}\parallel P(X)\parallel \mathit{Prod}))\\
\doteq & \nabla_{\{\mathit{ord}|\mathit{eo},\mathit{del}|\mathit{ao}\}}(\Gamma_{\{p_1|p_2\rightarrow\tau,p_3|p_4\rightarrow\tau\}}(\mathit{Recv}\mathbin{|\!\lfloor}(P(X)\parallel \mathit{Prod})+\\
 & \qquad\qquad (P(X)\mathbin{|\!\lfloor}\mathit{Prod}+\mathit{Prod}\mathbin{|\!\lfloor}P(X)+P(X)|\mathit{Prod})\mathbin{|\!\lfloor}\mathit{Recv}+\\
 & \qquad\qquad \mathit{Recv}|(P(X)\mathbin{|\!\lfloor}\mathit{Prod}+\mathit{Prod}\mathbin{|\!\lfloor}P(X)+P(X)|\mathit{Prod})))\\
\doteq & \delta+\delta+\delta+\sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s\wedge\langle i,s\rangle\mathrel{\in\!\!\!\!\bigcirc}X\rightarrow\langle \mathit{del}(i,s),\mathit{ao}(m,m-s)\rangle\cdot\\
 & \qquad \nabla_{\{\mathit{ord}|\mathit{eo},\mathit{del}|\mathit{ao}\}}(\Gamma_{\{p_1|p_2\rightarrow\tau,p_3|p_4\rightarrow\tau\}}(P(\langle i,s\rangle\ominus X)\parallel \mathit{Prod}\parallel \mathit{Recv}))+\\
 & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}}\langle \mathit{ord}(i,s),\mathit{eo}(n,n-s)\rangle\cdot\\
 & \qquad \nabla_{\{\mathit{ord}|\mathit{eo},\mathit{del}|\mathit{ao}\}}(\Gamma_{\{p_1|p_2\rightarrow\tau,p_3|p_4\rightarrow\tau\}}(\mathit{Recv}\parallel P(\langle i,s\rangle\oplus X)\parallel \mathit{Prod}))+\delta+\delta\\
\doteq & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s\wedge\langle i,s\rangle\mathrel{\in\!\!\!\!\bigcirc}X\rightarrow\langle \mathit{del}(i,s),\mathit{ao}(m,m-s)\rangle\cdot \mathit{OH}(\langle i,s\rangle\ominus X)+\\
 & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}}\langle \mathit{ord}(i,s),\mathit{eo}(n,n-s)\rangle\cdot \mathit{OH}(\langle i,s\rangle\oplus X)
\end{array}$$

$$\begin{array}{cl}
 & \mathit{DC}(n,m,X)\\
= & \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}(\mathit{OH}(X)\parallel \mathit{SH}\parallel E(n)\parallel A(m))))\\
\doteq & \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}((\mathit{OH}(X)|E(n))\mathbin{|\!\lfloor}(\mathit{SH}\parallel A(m)))))+\\
 & \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}((\mathit{OH}(X)|A(m))\mathbin{|\!\lfloor}(\mathit{SH}\parallel E(n)))))+\\
 & \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}((\mathit{SH}|E(n))\mathbin{|\!\lfloor}(\mathit{OH}(X)\parallel A(m)))))+\\
 & \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}((\mathit{SH}|A(m))\mathbin{|\!\lfloor}(\mathit{OH}(X)\parallel E(n)))))\\
\doteq & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}} \mathit{ord}(i,s)\cdot\\
 & \quad \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}(\mathit{OH}(\langle i,s\rangle\oplus X)\parallel \mathit{SH}\parallel E(n-s)\parallel A(m))))+\\
 & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s\wedge\langle i,s\rangle\mathrel{\in\!\!\!\!\bigcirc}X\rightarrow \mathit{del}(i,s)\cdot\\
 & \quad \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}(\mathit{OH}(\langle i,s\rangle\ominus X)\parallel \mathit{SH}\parallel E(n)\parallel A(m-s))))+\\
 & (n<\alpha)\rightarrow \mathit{supord}\cdot\\
 & \quad \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}(\mathit{OH}(X)\parallel \mathit{SH}\parallel E(n+\sigma)\parallel A(m))))+\\
 & \mathit{supdel}\cdot\\
 & \quad \nabla_{\{\mathit{ord},\mathit{del},\mathit{supord},\mathit{supdel}\}}(\Gamma_{\{e|\mathit{es}\rightarrow\tau,a|\mathit{as}\rightarrow\tau\}}(\Gamma_{\{e|\mathit{eo}\rightarrow\tau,a|\mathit{ao}\rightarrow\tau\}}(\mathit{OH}(X)\parallel \mathit{SH}\parallel E(n)\parallel A(m+\sigma))))\\
\doteq & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}} \mathit{ord}(i,s)\cdot \mathit{DC}(n-s,m,\langle i,s\rangle\oplus X)+\\
 & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s\wedge\langle i,s\rangle\mathrel{\in\!\!\!\!\bigcirc}X\rightarrow \mathit{del}(i,s)\cdot \mathit{DC}(n,m-s,\langle i,s\rangle\ominus X)+\\
 & (n<\alpha)\rightarrow \mathit{supord}\cdot \mathit{DC}(n+\sigma,m,X)+\\
 & \mathit{supdel}\cdot \mathit{DC}(n,m+\sigma,X)
\end{array}$$

$$
\begin{array}{ll}
 & Q(n,k,m,X) \\
= & \nabla_{\{ord,del\}}(\Gamma_{\{supord|si\to\tau,supdel|so\to\tau\}}(DC(n,m,X) \parallel S(k))) \\
\doteq & \nabla_{\{ord,del\}}(\Gamma_{\{supord|si\to\tau,supdel|so\to\tau\}}(DC(n,m,X) \lfloor\!\lfloor S(k) + S(k) \lfloor\!\lfloor DC(n,m,X) + DC(n,m,X)|S(k))) \\
\doteq & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}} ord(i,s)\cdot \nabla_{\{ord,del\}}(\Gamma_{\{supord|si\to\tau,supdel|so\to\tau\}}(DC(n-s,m,\langle i,s\rangle \oplus X) \parallel S(k)))+ \\
 & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s \wedge \langle i,s\rangle \mathrel{\textcircled{e}} X \to del(i,s)\cdot \\
 & \qquad\qquad \nabla_{\{ord,del\}}(\Gamma_{\{supord|si\to\tau,supdel|so\to\tau\}}(DC(n,m-s,\langle i,s\rangle \ominus X) \parallel S(k)))+ \\
 & \delta+ \\
 & n<\alpha \to \tau\cdot \nabla_{\{ord,del\}}(\Gamma_{\{supord|si\to\tau,supdel|so\to\tau\}}(DC(n+\sigma,m,X) \parallel S(k+1)))+ \\
 & k>0 \to \tau\cdot \nabla_{\{ord,del\}}(\Gamma_{\{supord|si\to\tau,supdel|so\to\tau\}}(DC(n,m+\sigma,X) \parallel S(k-1))) \\
\doteq & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{n:\mathbb{Z}} ord(i,s)\cdot Q(n-s,k,m,\langle i,s\rangle \oplus X)+ \\
 & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}}\sum_{m:\mathbb{N}} m>s \wedge \langle i,s\rangle \mathrel{\textcircled{e}} X \to del(i,s)\cdot Q(n,k,m-s,\langle i,s\rangle \ominus X)+ \\
 & n<\alpha \to \tau\cdot Q(n+\sigma,k+1,m,X)+ \\
 & k>0 \to \tau\cdot Q(n,k-1,m+\sigma,X)
\end{array}
$$

Process $Q(n,k,m,X)$ is convergent. A firing of any of these transitions will not disable any action that could occur before. We may thus assume that $k=0$ and that *deliv* fires whenever $k>0$ would occur, i.e. when *order* would fire. Similarly, *order* will fire whenever $n<\alpha$ would occur. This can only happen when *Recv* fires. So when *Recv* would fire causing $n<\alpha$, *order, deliv* will fire immediately after that; moreover they keep firing until $n \geq \alpha$. Modulo branching bisimilarity, *DCU* corresponds to $R(\alpha,\alpha,[])$, where R is specified as follows.

$$
\begin{array}{lll}
R(n:\mathbb{Z},m:\mathbb{N},X:\mathbb{B}(\mathcal{I}\times\mathbb{N})) & = & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}} ord(i,s)\cdot R(n-s+\mu,m+\mu,\langle i,s\rangle \oplus X)+ \\
 & & \sum_{i:\mathcal{I}}\sum_{s:\mathbb{N}} m>s \wedge \langle i,s\rangle \mathrel{\textcircled{e}} X \to del(i,s)\cdot R(n,m-s,\langle i,s\rangle \ominus X) \\
 & & \text{where } \mu = \max(0,-((n-s-\alpha)\div\sigma))
\end{array}
$$

We then prove the invariants $m-n=\sum_{\langle i,s\rangle \mathrel{\textcircled{e}} X} s * X(\langle i,s\rangle)$ and $n>\alpha$ for all reachable states $R(n,m,X)$. Thus the condition $m>s$ for the firing of *Prod* becomes redundant. Thus, we can remove the parameters n,m from the equations and obtain $DCU = S([])$, where

$$
\begin{array}{lll}
S(X:\mathbb{B}(\mathcal{I}\times\mathbb{N})) & = & \sum_{i:\mathcal{I},s:\mathbb{N}} ord(i,s)\cdot S(\langle i,s\rangle \oplus X)+ \\
 & & \langle i,s\rangle \mathrel{\textcircled{e}} X \to del(i,s)\cdot S(\langle i,s\rangle \ominus X)
\end{array}
$$

This the specification of a place. So any order will eventually be delivered.

Although we are now convinced that $DCU = S([])$, we only have given a reasoning why this should be true. We now prove this by proving the bisimilarity of $Q(n,k,m,X)$ and $S(X)$ with the cones and foci method, with invariant $m-n+k\sigma = \sum_{\langle i,s\rangle \mathrel{\textcircled{e}} X} s * X(\langle i,s\rangle)$ of Q and state mapping h defined by $h(\langle n,k,m,X\rangle = \langle X\rangle$.

1. $Q(n,k,m.,X)$ is convergent with the smallest well-founded order that satisfies $(n<\alpha \equiv \langle n+\sigma,k+1,m,X\rangle < \langle n,k,m,X\rangle) \wedge (k>0 \equiv \langle n,k-1,m+\sigma,X\rangle < \langle n,k,m,X\rangle)$ (which is well-founded as in any chain $2*(\alpha-n)+k$ decreases and has a lower bound).

2. $n<\alpha \Rightarrow h(\langle n,k,m,X\rangle) = h(\langle n+\sigma,k+1,m,X\rangle)$ and $k>0 \Rightarrow h(\langle n,k,m,X\rangle) = h(\langle n,k-1,m+\sigma,X\rangle)$, which is trivially true.

3. $true \Rightarrow true$ and $m>0 \wedge \langle i,s\rangle \mathrel{\textcircled{e}} X \Rightarrow \langle i,s\rangle \mathrel{\textcircled{e}} X$, trivial.

4. $n\geq\alpha \wedge k=0 \wedge true \Rightarrow true$ and $n\geq\alpha \wedge k=0 \wedge \langle i,s\rangle \mathrel{\textcircled{e}} X \Rightarrow m>0 \wedge \langle i,s\rangle \mathrel{\textcircled{e}} X$, trivial with $n\geq\alpha \wedge k=0 \Rightarrow m-\alpha \geq \sum_{\langle i,s\rangle \mathrel{\textcircled{e}} X} s * X(\langle i,s\rangle) \Rightarrow m>0$

5. Trivial, as the action parameters do not depend on m,k,n or X.

6. $true \Rightarrow h(\langle n-s,k,m,\langle i,s\rangle \oplus X) = \langle\langle i,s\rangle \oplus X\rangle\rangle$ and $\langle i,s\rangle \mathrel{\textcircled{e}} X \Rightarrow h(\langle n-s,k,m,\langle i,s\rangle \ominus X) = \langle\langle i,s\rangle \ominus X\rangle\rangle$, trivial.

11 Example: Guarded Command Language

Given the following *multiprogram* we wish to determine its termination behaviour by transforming it to a process algebra expression.

```
|[ const $val_b, val_c$ : bool;
   var $b, c$ : bool;
|  { $b = val_b \wedge c = val_c$ }
   |[ if $b$ → skip
      fi
   ;  $c := true$
   ]|
||
   |[ if $c$ → skip
      fi
   ;  $b := true$
   ]|
]|
```

First we translate the (global) variables in the following way:

$$VarB(id:\mathcal{I}, val:B) = \overline{r_B}(id, val)\cdot VarB(id, val) + \textstyle\sum_{b:B}\overline{s_B}(id,b)\cdot VarB(id,b)$$

Next we translate the combination of the **if** and the assignment by process S.

$$S(id_1:\mathcal{I}, id_2:\mathcal{I}) = r_B(id_1, true)\cdot s_B(id_2, true)$$

The full translation then becomes as follows:

$$\begin{array}{l} P(val_b:B, val_c:B) = \nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel S(id_b, id_c)\parallel S(id_c, id_b)))\\ \qquad\text{with } V=\{\}\text{ and } C=\{\overline{r_B}|r_B\to\tau, \overline{s_B}|s_B\to\tau\}\end{array}$$

Now we can calculate with this expression:

$$\begin{array}{cl}
 & P(val_b, val_c)\\
= & \nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel S(id_b, id_c)\parallel S(id_c, id_b)))\\
\doteq & val_b=\mathbf{t}\to\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel s_B(id_c, true)\parallel S(id_c, id_b)))+\\
 & val_c=\mathbf{t}\to\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel S(id_b, id_c)\parallel s_B(id_b, true)))+\\
 & val_b=\mathbf{t}\ \wedge val_c=\mathbf{t}\to\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel s_B(id_c, true)\parallel s_B(id_b, true)))\\
\doteq & val_b\to\tau\cdot(\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, true)\parallel S(id_c, id_b)))+\\
 & \qquad\qquad val_c\to\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel s_B(id_c, true)\parallel s_B(id_b, true))))+\\
 & val_c\to\tau\cdot(\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, true)\parallel VarB(id_c, val_c)\parallel S(id_b, id_c)))+\\
 & \qquad\qquad val_b\to\tau\cdot\nabla_V(\Gamma_C(VarB(id_b, val_b)\parallel VarB(id_c, val_c)\parallel s_B(id_c, true)\parallel s_B(id_b, true))))+\\
 & val_b\wedge val_c\to\tau\cdot(\tau\cdot\tau+\tau\cdot\tau)\\
\doteq & val_b\to\tau\cdot(\tau\cdot\tau\cdot\tau+val_c\to\tau\cdot(\tau\cdot\tau+\tau\cdot\tau))+\\
 & val_c\to\tau\cdot(\tau\cdot\tau\cdot\tau+val_b\to\tau\cdot(\tau\cdot\tau+\tau\cdot\tau))+\\
 & val_b\wedge val_c\to\tau\\
\doteq & val_b\to\tau\cdot(\tau+val_c\to\tau)+\\
 & val_c\to\tau\cdot(\tau+val_b\to\tau)+\\
 & val_b\wedge val_c\to\tau\\
\doteq & val_b\to\tau\cdot(\ \mathbf{t}\ \vee val_c\to\tau)+\\
 & val_c\to\tau\cdot(\ \mathbf{t}\ \vee val_d\to\tau)+\\
 & val_b\wedge val_c\to\tau\\
\doteq & val_b\to\tau+val_c\to\tau+val_b\wedge val_c\to\tau\\
\doteq & val_b\vee val_c\vee(val_b\wedge val_c)\to\tau\\
\doteq & val_b\vee val_c\to\tau
\end{array}$$

As could be expected, the multiprogram only terminates if val_b or val_c holds (or both). Otherwise both subprograms deadlock and thus the multiprogram itself as well.

12 Alphabet axioms

As the linearisation of parallel expressions has the tendency to significantly increase the amount of memory needed to store resulting expressions and often such expressions are enclosed by for instance restriction operators, one wishes to be able to remove as much of the (sub)expressions as possible before elimination of parallel operators. To help this process we give a list of alphabet axioms, which can be used to easily *push* certain operators deeper into an expression to (partially) effectuate its behaviour and possibly limiting the increase in memory needed for linearisation.

Definition 12.1. Let $p, p' \in T_{pc}$ and $m \in \mathbb{B}(A)$. We define the alphabet $\alpha_\nu(p)$ of a process p by $(p \xrightarrow{m} \checkmark \Rightarrow \mu(m) \in \alpha_\nu(p)) \wedge (p \xrightarrow{m} p' \Rightarrow \{\mu(m)\} \cup \alpha_\nu(p') \subseteq \alpha_\nu(p))$.

Note that the above definition is in terms of the semantics. This suits us when proving soundness, but during linearisation one probably wants an axiomatic definition. Although we do not give such a definition here, we do want to note that in practice it is probably well acceptable to use an over-approximation for α_ν (i.e. some $\widetilde{\alpha_\nu}$ with $\forall_t(\alpha_\nu(t) \subseteq \widetilde{\alpha_\nu}(t))$), as if the axioms presented below hold for such an over-approximation, they will certainly hold for α_ν.

Definition 12.2. Let $a, b \in \mathcal{N}_A$, $v, w \in \mathbb{B}(\mathcal{N}_A)$ and $V \subseteq \mathbb{B}(\mathcal{N}_A)$. We define the set $\mathcal{N}(V)$ of actions in a set of bags V as follows:

$$\mathcal{N}(V) = \{a \mid a \in v \wedge v \in V\}$$

Definition 12.3. Let $a, b \in \mathbb{B}(\mathcal{N}_A)$ and $V \subseteq \mathbb{B}(\mathcal{N}_A)$. We define the set $\Downarrow(V)$ of subbags of bags in set V by $\Downarrow(V) = \{b \mid a \in V \wedge b \subseteq a\}$.

Definition 12.4. Let S, T be sets and $F : S \to T$. Also, let $s \in S$ and $t \in T$. We define the *dom*(F) and *rng*(F) as follows:

$$\begin{array}{lcl} dom(F) & = & \{s \mid \exists_{t \in T}(\langle s,t\rangle \in F)\} \\ rng(F) & = & \{t \mid \exists_{s \in S}(\langle s,t\rangle \in F)\} \end{array}$$

Note we write $S \cap T$ and $S \cup T$ while we actually mean a (syntactic) set U such that $[\![U]\!] = [\![S]\!] \cap [\![T]\!]$ or $[\![U]\!] = [\![S]\!] \cup [\![T]\!]$ respectively.

With these definitions we have the following alphabet axioms:

*VA*1	$\nabla_V(x) \doteq x \quad \mathit{if}\ \alpha_\nu(x) \subseteq [\![V]\!]$
*VA*2	$\nabla_V(x) \doteq \delta \quad \mathit{if}\ [\![V]\!] \cap \alpha_\nu(x) = \emptyset$
*VA*3	$\nabla_V(\nabla_{V'}(x)) \doteq \nabla_{V \cap V'}(x)$
*VA*4	$\nabla_V(x \parallel y) \doteq \nabla_V(x \parallel \nabla_{V'}(y)) \quad \mathit{if}\ \Downarrow([\![V]\!]) \subseteq [\![V']\!]$
*CA*1	$\Gamma_C(x) \doteq x \quad \mathit{if}\ dom([\![C]\!]) \cap \Downarrow(\alpha_\nu(x)) = \emptyset$
*CA*2	$\Gamma_C(\Gamma_{C'}(x)) \doteq \Gamma_{C \cup C'}(x) \quad \mathit{if}\ \mathcal{N}(dom([\![C]\!])) \cap \mathcal{N}(dom([\![C']\!])) = \emptyset \wedge \mathcal{N}(dom([\![C]\!])) \cap rng([\![C']\!]) = \emptyset$
*CA*3	$\Gamma_C(x \parallel y) \doteq x \parallel \Gamma_C(y) \quad \mathit{if}\ \Downarrow(dom([\![C]\!])) \cap \Downarrow(\alpha_\nu(x)) = \emptyset$
*CA*4	$\Gamma_C(x \parallel y) \doteq \Gamma_C(x \parallel \Gamma_C(y)) \quad \mathit{if}\ \mathcal{N}(dom([\![C]\!])) \cap rng([\![C]\!]) = \emptyset$
*DA*1	$\partial_H(x) \doteq x \quad \mathit{if}\ [\![H]\!] \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset$
*DA*2	$\partial_H(x) \doteq \delta \quad \mathit{if}\ \forall_{v \in \alpha_\nu(x)}(v \cap [\![H]\!] \neq \emptyset)$

DA3	$\partial_H(\partial_{H'}(x)) \doteq \partial_{H\cup H'}(x)$
DA4	$\partial_H(x \parallel y) \doteq \partial_H(x) \parallel \partial_H(y)$
TA1	$\tau_I(x) \doteq x \quad \mathit{if}\ \llbracket I\rrbracket \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset$
TA3	$\tau_I(\tau_{I'}(x)) \doteq \tau_{I\cup I'}(x)$
TA4	$\tau_I(x \parallel y) \doteq \tau_I(x) \parallel \tau_I(y)$
RA1	$\rho_R(x) \doteq x \quad \mathit{if}\ \mathit{dom}(\llbracket R\rrbracket) \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset$
RA2	$\rho_R(\rho_{R'}(x)) \doteq \rho_{R\cup R'}(x) \quad \mathit{if}\ \mathit{dom}(\llbracket R\rrbracket) \cap \mathit{dom}(\llbracket R'\rrbracket) = \emptyset \wedge \mathit{dom}(\llbracket R\rrbracket) \cap \mathit{rng}(\llbracket R'\rrbracket) = \emptyset$
RA3	$\rho_R(\rho_{R'}(x)) \doteq \rho_{R''}(x) \quad \mathit{if}\ \llbracket R''\rrbracket = \{\langle a,b\rangle \mid ((\langle a,b\rangle \in \llbracket R\rrbracket \wedge a \notin (\mathit{dom}(\llbracket R'\rrbracket) \cup \mathit{rng}(\llbracket R'\rrbracket)))\vee$ $((\langle c,b\rangle \in \llbracket R\rrbracket \wedge \langle a,c\rangle \in \llbracket R'\rrbracket)\vee$ $((\langle a,b\rangle \in \llbracket R'\rrbracket \wedge b \notin \mathit{dom}(\llbracket R\rrbracket))\}$
RA4	$\rho_R(x \parallel y) \doteq \rho_R(x) \parallel \rho_R(y)$
VC1	$\nabla_V(\Gamma_C(x)) \doteq \nabla_V(\Gamma_C(\nabla_{V'}(x))) \quad \mathit{if}\ \llbracket V'\rrbracket = \{v \oplus w \mid \gamma(v, \llbracket C\rrbracket) \oplus w \in \llbracket V\rrbracket\}$
VC2	$\Gamma_C(\nabla_V(x)) \doteq \nabla_V(x) \quad \mathit{if}\ \mathit{dom}(C) \cap \Downarrow(V) = \emptyset$
VD1	$\nabla_V(\partial_H(x)) \doteq \partial_H(\nabla_V(x))$
VD2	$\nabla_V(\partial_H(x)) \doteq \nabla_{V'}(x) \quad \mathit{if}\ \llbracket V'\rrbracket = \{v \mid v \in \llbracket V\rrbracket \wedge \mathcal{N}(\{v\}) \cap \llbracket H\rrbracket = \emptyset\}$
VD3	$\partial_H(\nabla_V(x)) \doteq \nabla_{V'}(x) \quad \mathit{if}\ \llbracket V'\rrbracket = \{v \mid v \in \llbracket V\rrbracket \wedge \mathcal{N}(\{v\}) \cap \llbracket H\rrbracket = \emptyset\}$
VT	$\nabla_V(\tau_I(x)) \doteq \tau_I(\nabla_{V'}(x)) \quad \mathit{if}\ \llbracket V'\rrbracket = \{v \mid \theta(v, \llbracket I\rrbracket) \in \llbracket V\rrbracket\}$
VR	$\nabla_V(\rho_R(x)) \doteq \rho_R(\nabla_{V'}(x)) \quad \mathit{if}\ \llbracket V'\rrbracket = \{v \mid (\llbracket R\rrbracket \bullet v) \in \llbracket V\rrbracket\}$
CD1	$\partial_H(\Gamma_C(x)) \doteq \Gamma_C(\partial_H(x)) \quad \mathit{if}\ (\mathcal{N}(\mathit{dom}(C)) \cup \mathit{rng}(C)) \cap H = \emptyset$
CD2	$\Gamma_C(\partial_H(x)) \doteq \partial_H(x) \quad \mathit{if}\ \mathcal{N}(\mathit{dom}(C)) \subseteq H$
CT1	$\tau_I(\Gamma_C(x)) \doteq \Gamma_C(\tau_I(x)) \quad \mathit{if}\ (\mathcal{N}(\mathit{dom}(C)) \cup \mathit{rng}(C)) \cap I = \emptyset$
CT2	$\Gamma_C(\tau_I(x)) \doteq \tau_I(x) \quad \mathit{if}\ \mathcal{N}(\mathit{dom}(C)) \subseteq I$
CR1	$\rho_R(\Gamma_C(x)) \doteq \Gamma_C(\rho_R(x)) \quad \mathit{if}\ \mathit{dom}(R) \cap \mathit{rng}(C) = \mathit{dom}(R) \cap \mathcal{N}(\mathit{dom}(C)) = \mathit{rng}(R) \cap \mathcal{N}(\mathit{dom}(C)) = \emptyset$
CR2	$\Gamma_C(\rho_R(x)) \doteq \rho_R(x) \quad \mathit{if}\ \mathcal{N}(\mathit{dom}(C)) \subseteq \mathit{dom}(R) \wedge \mathcal{N}(\mathit{dom}(C)) \cap \mathit{rng}(R) = \emptyset$
DT	$\partial_H(\tau_I(x)) \doteq \tau_I(\partial_H(x)) \quad \mathit{if}\ \llbracket I\rrbracket \cap \llbracket H\rrbracket = \emptyset$
DR	$\partial_H(\rho_R(x)) \doteq \rho_R(\partial_{H'}(x)) \quad \mathit{if}\ \llbracket H'\rrbracket = \{v \mid (\llbracket R\rrbracket \bullet v) \in \llbracket H\rrbracket\}$
TR	$\tau_I(\rho_R(x)) \doteq \rho_R(\tau_{I'}(x)) \quad \mathit{if}\ \llbracket I\rrbracket = \{\llbracket R\rrbracket^+(a) \mid a \in \llbracket I'\rrbracket\}$

Table 3: GenSpect Alphabet Axioms

To prove the soundness of these alphabet axioms, we use a different approach as before, except for the axioms containing a parallel operator. As the operators used in these axioms are all defined in a similar way, namely:

$$\frac{x \xrightarrow{m} \surd}{O(x) \xrightarrow{f(m)} \surd}\ C(m) \qquad \frac{x \xrightarrow{m} x'}{O(x) \xrightarrow{f(m)} O(x')}\ C(m)$$

If we now wish to prove an axiom $O_1(\ldots O_n(x)\ldots) \doteq O_{n+1}(\ldots O_{n'}(x)\ldots)$, we get the following:

- $O_1(\ldots O_n(x)\ldots) \xrightarrow{m} \surd$, which means that $O_2(\ldots O_n(x)\ldots) \xrightarrow{m_1} \surd \wedge C_1(m_1) \wedge m = f_1(m_1) \ldots x \xrightarrow{m_n} \surd \wedge C_n(m_n) \wedge m_{n-1} = f_n(m_n)$ and therefore $C_{n+1}(m_n) \wedge \ldots \wedge C_{n'}(f_{n'-1}(\ldots f_{n+1}(m_n)\ldots))$ and $m = f'_n(\ldots f_{n+1}(m_n)\ldots)$ and $O_{n+1}(\ldots O_{n'}(x)\ldots) \xrightarrow{m} \surd$
- Etc.

So what really has to be proven is that for each m, such that $x \xrightarrow{m} \surd \vee x \xrightarrow{m} x'$, it holds that $C_n(m) \wedge \ldots \wedge C_1(f_2(\ldots f_n(m))) \equiv C_{n'}(m) \wedge \ldots \wedge C_{n+1}(f_{n+2}(\ldots f_{n'}(m)))$ and $C_n(m) \wedge \ldots \wedge C_1(f_2(\ldots f_n(m))) \Rightarrow f_1(\ldots f_n(m)) = f_{n+1}(\ldots f_{n'}(m))$.The proofs for the alphabet axioms are given in Appendix C.

13 Future work

Although we have introduced a process algebra and have proven its axiomatisation to be sound and (relatively) complete, it might not be precisely what one wants in practice. Besides the addition of axioms for recursive specifications and perhaps completeness proofs for these specifications as well as for the use of (rooted) branching bisimulation, we consider the addition of the empty process ϵ (or *skip*) a desired step, as it is a quite natural and practical building block. Unfortunately, this will require all soundness and completeness proofs to be rewritten. Another highly preferred addition is that of time. Here, a choice will have to be made between relative and absolute time, and should only require additions to the already given proofs. Also, one will have to think about the exact meaning of a multiaction in a timed algebra.

Of course, for this algebra to be really of practical use, a number of tools will have to be written, such as a lineariser, a state space generator and tools to convert higher level modelling languages to equivalent algebraic specifications.

A Soundness proofs

The structure of each of these proofs is the same, namely:

- A relation R is given for the axiom. For an axiom $t \doteq u$, the relation R is the minimal relation that satisfies $tRu \wedge pRp \wedge (qRr \Leftrightarrow rRq) \wedge E$, for all $p, q, r \in T_{pc}$ and for all instantiations of process variables in t and u. E is *true*, unless it is explicitly defined otherwise in the subsection.
- For each conjunct of the definition of R it is shown that the properties that define a bisimulation hold.

It is clear that if this is done, R is proven to be a bisimulation relation.

To shorten the proofs and get rid of trivial facts, that would otherwise be repeated in every subsection, we do not define R explicitly (as it is always of the form given above) or prove the conjuncts pRp and $qRr \Leftrightarrow rRq$ (and E, if it is *true*). The proofs of these conjuncts are trivial and therefore not given at all.

When we say a process cannot terminate in any of the subsections, we mean that there is no m for which $\xrightarrow{m} \surd$ holds for that process. (So it could very well be that the process can indeed terminate, but not in one "step".)

A.1 $MA1$ $a \doteq \langle a \rangle$

- $a \xrightarrow{m} \surd$, which means that $m = [a]$ and $\langle a \rangle \xrightarrow{[a]} \surd$
- $\langle a \rangle \xrightarrow{m} \surd$, which is the same as the previous case
- $a \xrightarrow{m} p$, which is not possible
- $\langle a \rangle \xrightarrow{m} p$, which is not possible

A.2 $MA2$ $\tau \doteq \langle\rangle$

This proof is similar to that of axiom $MA2$ with $m = []$.

A.3 $MA3$ $\langle \vec{a}, \vec{b} \rangle \doteq \langle \vec{b}, \vec{a} \rangle$

Both $\langle \vec{a}, \vec{b} \rangle$ and $\langle \vec{b}, \vec{a} \rangle$ cannot do any transition. They can only terminate with $[\![\langle \vec{a}, \vec{b} \rangle]\!]$ and $[\![\langle \vec{b}, \vec{a} \rangle]\!]$ resp., which are equal.

A.4 $A1$ $x + y \doteq y + x$

- $p + q \xrightarrow{m} \surd$, which means that
  - $p \xrightarrow{m} \surd$, which means that $q + p \xrightarrow{m} \surd$, or
  - $q \xrightarrow{m} \surd$, which means that $q + p \xrightarrow{m} \surd$
- $q + p \xrightarrow{m} \surd$, symmetrical to the prove above
- $p + q \xrightarrow{m} r$, which means that
  - $p \xrightarrow{m} r$, which means that $q + p \xrightarrow{m} r$, with rRr, or
  - $q \xrightarrow{m} r$, which means that $q + p \xrightarrow{m} r$, with rRr
- $q + p \xrightarrow{m} r$, symmetrical to the prove above

A.5 $A2 \quad x + (y + z) \doteq (x + y) + z$

- $p + (q + r) \xrightarrow{m} \surd$, which means that
  - $p \xrightarrow{m} \surd$, which means that $p + q \xrightarrow{m} \surd$ and $(p + q) + r \xrightarrow{m} \surd$, or
  - $q + r \xrightarrow{m} \surd$, which means that
    - $q \xrightarrow{m} \surd$, which means that $p + q \xrightarrow{m} \surd$ and $(p + q) + r \xrightarrow{m} \surd$, or
    - $r \xrightarrow{m} \surd$, which means that $(p + q) + p \xrightarrow{m} \surd$
- $(p + q) + r \xrightarrow{m} \surd$, symmetrical to the prove above
- $p + (q + r) \xrightarrow{m} r'$, which means that
  - $p \xrightarrow{m} r'$, which means that $p + q \xrightarrow{m} r'$ and $(p + q) + r \xrightarrow{m} r'$, with $r'Rr'$, or
  - $q + r \xrightarrow{m} r'$, which means that
    - $q \xrightarrow{m} r'$, which means that $p + q \xrightarrow{m} r'$ and $(p + q) + r \xrightarrow{m} r'$, with $r'Rr'$, or
    - $r \xrightarrow{m} r'$, which means that $(p + q) + r \xrightarrow{m} r'$, with $r'Rr'$
- $(p + q) + r \xrightarrow{m} r'$, symmetrical to the prove above

A.6 $A3 \quad x + x \doteq x$

- $p + p \xrightarrow{m} \surd$, which means that $p \xrightarrow{m} \surd$
- $p \xrightarrow{m} \surd$, which means that $p + p \xrightarrow{m} \surd$
- $p + p \xrightarrow{m} p'$, which means that $p \xrightarrow{m} p'$, with $p'Rp'$
- $p \xrightarrow{m} p'$, which means that $p + p \xrightarrow{m} p'$, with $p'Rp'$

A.7 $A4 \quad (x + y) \cdot z \doteq x \cdot z + y \cdot z$

- $(p + q) \cdot r \xrightarrow{m} \surd$, which is not possible
- $p \cdot r + q \cdot r \xrightarrow{m} \surd$, which means that
  - $p \cdot r \xrightarrow{m} \surd$, which is not possible, or
  - $q \cdot r \xrightarrow{m} \surd$, which is not possible
- $(p + q) \cdot r \xrightarrow{m} r'$, which means that
  - $p + q \xrightarrow{m} \surd \wedge r' = r$, which means that
    - $p \xrightarrow{m} \surd$, which means that $p \cdot r \xrightarrow{m} r$ and $p \cdot r + q \cdot r \xrightarrow{m} r$, with $r'Rr$, or
    - $q \xrightarrow{m} \surd$, which means that $q \cdot r \xrightarrow{m} r$ and $p \cdot r + q \cdot r \xrightarrow{m} r$, with $r'Rr$, or
  - $p + q \xrightarrow{m} q' \wedge r' = q' \cdot r$ which means that
    - $p \xrightarrow{m} q'$, which means that $p \cdot r \xrightarrow{m} q' \cdot r$ and $p \cdot r + q \cdot r \xrightarrow{m} q' \cdot r$, with $r'Rq' \cdot r$, or
    - $q \xrightarrow{m} q'$, which means that $q \cdot r \xrightarrow{m} q' \cdot r$ and $p \cdot r + q \cdot r \xrightarrow{m} q' \cdot r$, with $r'Rq' \cdot r$
- $p \cdot r + q \cdot r \xrightarrow{m} r'$, which means that
  - $p \cdot r \xrightarrow{m} r'$, which means that
    - $p \xrightarrow{m} \surd \wedge r' = r$, which means that $p + q \xrightarrow{m} \surd$ and $(p + q) \cdot r \xrightarrow{m} r$, with $r'Rr$, or

    - $p \xrightarrow{m} p' \wedge r' = p' \cdot r$, which means that $p + q \xrightarrow{m} p'$ and $(p+q) \cdot r \xrightarrow{m} p' \cdot r$, with $r' R p' \cdot r$, or
  - $q \cdot r \xrightarrow{m} r'$, which means that
    - $q \xrightarrow{m} \surd \wedge r' = r$, which means that $p + q \xrightarrow{m} \surd$ and $(p+q) \cdot r \xrightarrow{m} r$, with $r' R r$, or
    - $q \xrightarrow{m} q' \wedge r' = q' \cdot r$, which means that $p + q \xrightarrow{m} q'$ and $(p+q) \cdot r \xrightarrow{m} q' \cdot r$, with $r' R q' \cdot r$

A.8 *A5* $(x \cdot y) \cdot z \doteq x \cdot (y \cdot z)$

- $(p \cdot q) \cdot r \xrightarrow{m} \surd$, which is not possible
- $p \cdot (q \cdot r) \xrightarrow{m} \surd$, which is not possible
- $(p \cdot q) \cdot r \xrightarrow{m} r'$, which means that
  - $p \cdot q \xrightarrow{m} \surd$, which is not possible
  - $p \cdot q \xrightarrow{m} q' \wedge r' = q' \cdot r$, which means that
    - $p \xrightarrow{m} \surd \wedge q' = q$, which means that $p \cdot (q \cdot r) \xrightarrow{m} q \cdot r$, with $r' R q \cdot r$, or
    - $p \xrightarrow{m} p' \wedge q' = p' \cdot q$, which means that $p \cdot (q \cdot r) \xrightarrow{m} p' \cdot (q \cdot r)$, with $r' R (p' \cdot q) \cdot r$

A.9 *A6* $x + \delta \doteq x$

- $p + \delta \xrightarrow{m} \surd$, which means that $p \xrightarrow{m} \surd$
- $p \xrightarrow{m} \surd$, which means that $p + \delta \xrightarrow{m} \surd$
- $p + \delta \xrightarrow{m} p'$, which means that $p \xrightarrow{m} p'$, with $p' R p'$
- $p \xrightarrow{m} p'$, which means that $p + \delta \xrightarrow{m} p'$, with $p' R p'$

A.10 *A7* $\delta \cdot x \doteq \delta$

Both $\delta \cdot p$ and δ cannot terminate or do any transition.

A.11 *C1* $\boldsymbol{t} \rightarrow x \doteq x$

- $\boldsymbol{t} \rightarrow p \xrightarrow{m} \surd$, which means that $p \xrightarrow{m} \surd$
- $p \xrightarrow{m} \surd$, which means that $\boldsymbol{t} \rightarrow p \xrightarrow{m} \surd$
- $\boldsymbol{t} \rightarrow p \xrightarrow{m} p'$, which means that $p \xrightarrow{m} p'$, with $p' R p'$
- $p \xrightarrow{m} p'$, which means that $\boldsymbol{t} \rightarrow p \xrightarrow{m} p'$, with $p' R p'$

A.12 *C2* $\boldsymbol{f} \rightarrow x \doteq \delta$

Both $\boldsymbol{f} \rightarrow p$ and δ cannot terminate or make any transition.

A.13 *SUM*1 $\sum_{d:D} x \doteq x$

Note that p cannot contain d as it is a substitution for x.

- $\sum_{d:D} p \xrightarrow{m} \surd$, which means that $\exists_{e:D} p[e/d] \xrightarrow{m} \surd$ and $p \xrightarrow{m} \surd$
- $p \xrightarrow{m} \surd$, which means that $\exists_{e:D} p[e/d] \xrightarrow{m} \surd$ and $\sum_{d:D} p \xrightarrow{m} \surd$
- $\sum_{d:D} p \xrightarrow{m} p'$, which means that $\exists_{e:D} p[e/d] \xrightarrow{m} p'$ and $p \xrightarrow{m} p'$, with $p'Rp'$
- $p \xrightarrow{m} p'$, which means that $\exists_{e:D} p[e/d] \xrightarrow{m} p'$ and $\sum_{d:D} p \xrightarrow{m} p'$, with $p'Rp'$

A.14 *SUM*3 $\sum_{d:D} p \doteq \sum_{d:D} p + p[e/d]$ *with* $e \in D$

Assuming $e \in D$.

- $\sum_{d:D} p \xrightarrow{m} \surd$, which means that $\sum_{d:D} p + p[e/d] \xrightarrow{m} \surd$
- $\sum_{d:D} p + p[e/d] \xrightarrow{m} \surd$, which means that
  - $\sum_{d:D} p \xrightarrow{m} \surd$, or
  - $p[e/d] \xrightarrow{m} \surd$, which means that $\sum_{d:D} p \xrightarrow{m} \surd$
- $\sum_{d:D} p \xrightarrow{m} p'$, which means that $\sum_{d:D} p + p[e/d] \xrightarrow{m} p'$, with $p'Rp'$
- $\sum_{d:D} p + p[e/d] \xrightarrow{m} p'$, which means that
  - $\sum_{d:D} p \xrightarrow{m} p'$, with $p'Rp'$, or
  - $p[e/d] \xrightarrow{m} p'$, which means that $\sum_{d:D} p \xrightarrow{m} p'$, with $p'Rp'$

A.15 *SUM*4 $\sum_{d:D}(p + q) \doteq \sum_{d:D} p + \sum_{d:D} q$

- $\sum_{d:D}(p+q) \xrightarrow{m} \surd$, which means that $\exists_{e:D}((p[e/d] + q[e/d]) \xrightarrow{m} \surd)$ and
  - $p[e/d] \xrightarrow{m} \surd$, which means $\sum_{d:D} p \xrightarrow{m} \surd$ and $\sum_{d:D} p + \sum_{d:D} q \xrightarrow{m} \surd$, or
  - $q[e/d] \xrightarrow{m} \surd$, which means $\sum_{d:D} q \xrightarrow{m} \surd$ and $\sum_{d:D} p + \sum_{d:D} q \xrightarrow{m} \surd$
- $\sum_{d:D} p + \sum_{d:D} q \xrightarrow{m} \surd$, which means that
  - $\sum_{d:D} p \xrightarrow{m} \surd$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m} \surd)$ and $p[e/d] + q[e/d] \xrightarrow{m} \surd$ and $\sum_{d:D}(p + q) \xrightarrow{m} \surd$, or
  - $\sum Q \xrightarrow{m} \surd$, which means that $\exists_{e:D}(q[e/d] \xrightarrow{m} \surd)$ and $p[e/d] + q[e/d] \xrightarrow{m} \surd$ and $\sum_{d:D}(p + q) \xrightarrow{m} \surd$
- $\sum_{d:D}(p+q) \xrightarrow{m} p'$, which means that $\exists_{e:D}((p[e/d] + q[e/d]) \xrightarrow{m} p')$ and
  - $p[e/d] \xrightarrow{m} p'$, which means $\sum_{d:D} p \xrightarrow{m} p'$ and $\sum_{d:D} p + \sum_{d:D} q \xrightarrow{m} p'$, with $p'Rp'$, or
  - $q[e/d] \xrightarrow{m} p'$, which means $\sum_{d:D} q \xrightarrow{m} p'$ and $\sum_{d:D} p + \sum_{d:D} q \xrightarrow{m} p'$, with $p'Rp'$
- $\sum_{d:D} p + \sum_{d:D} q \xrightarrow{m} p'$, which means that
  - $\sum_{d:D} p \xrightarrow{m} p'$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m} p')$ and $p[e/d] + q[e/d] \xrightarrow{m} p'$ and $\sum_{d:D}(p + q) \xrightarrow{m} p'$, with $p'Rp'$, or
  - $\sum_{d:D} q \xrightarrow{m} p'$, which means that $\exists_{e:D}(q[e/d] \xrightarrow{m} p')$ and $p[e/d] + q[e/d] \xrightarrow{m} p'$ and $\sum_{d:D}(p + q) \xrightarrow{m} p'$, with $p'Rp'$

A.16 *SUM5* $(\sum_{d:D} p) \cdot y \doteq \sum_{d:D}(p \cdot y)$

Note that q cannot contain d as it is a substitution for y.

- Both $(\sum_{d:D} p) \cdot q$ and $\sum_{d:D}(p \cdot q)$ cannot terminate
- $(\sum_{d:D} p) \cdot q \xrightarrow{m} r$, which means that
  - $\sum_{d:D} p \xrightarrow{m} \surd \wedge r = q$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m} \surd)$ and $p[e/d] \cdot q[e/d] \xrightarrow{m} q$ and $\sum_{d:D}(p \cdot q) \xrightarrow{m} q$, with rRq, or
  - $\sum_{d:D} p \xrightarrow{m} p' \wedge r = p' \cdot q$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m} p')$ and $p[e/d] \cdot q[e/d] \xrightarrow{m} p' \cdot q$ and $\sum_{d:D}(p \cdot q) \xrightarrow{m} p' \cdot q$, with $rRp' \cdot q$
- $\sum_{d:D}(p \cdot q) \xrightarrow{m} r$, which means that $\exists_{e:D}((p[e/d] \cdot q[e/d]) \xrightarrow{m} r)$ and
  - $p[e/d] \xrightarrow{m} \surd \wedge r = q$, which means that $\sum_{d:D} p \xrightarrow{m} \surd$ and $(\sum_{d:D} p) \cdot q \xrightarrow{m} q$, with rRx, or
  - $p[e/d] \xrightarrow{m} r' \wedge r = r' \cdot q$, which means that $\sum_{d:D} p \xrightarrow{m} r'$ and $(\sum_{d:D} p) \cdot q \xrightarrow{m} r' \cdot q$, with $rRr' \cdot q$

A.17 *SUM6* $(\sum_{d:D} p) \mathbin{\|\!\_} y \doteq \sum_{d:D}(p \mathbin{\|\!\_} y)$

Note that q cannot contain d as it is a substitution for y.

- Both $(\sum_{d:D} p) \mathbin{\|\!\_} q$ and $\sum_{d:D}(p \mathbin{\|\!\_} q)$ cannot terminate
- $(\sum_{d:D} p) \mathbin{\|\!\_} q \xrightarrow{m} r$, which means that
  - $\sum_{d:D} p \xrightarrow{m} \surd \wedge r = q$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m} \surd)$ and $p[e/d] \mathbin{\|\!\_} q[e/d] \xrightarrow{m} q$ and $\sum_{d:D}(p \mathbin{\|\!\_} q) \xrightarrow{m} q$, with rRq, or
  - $\sum_{d:D} p \xrightarrow{m} p' \wedge r = p' \parallel q$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m} p'') \wedge p' = p''$ and $p[e/d] \mathbin{\|\!\_} q[e/d] \xrightarrow{m} p'' \parallel q$ and $\sum_{d:D}(p \mathbin{\|\!\_} q) \xrightarrow{m} p'' \parallel q$, with $rRp'' \parallel q$
- $\sum_{d:D}(p \mathbin{\|\!\_} q) \xrightarrow{m} r$, which means that $\exists_{e:D}(p[e/d] \mathbin{\|\!\_} q[e/d] \xrightarrow{m} r') \wedge r = z'$ and
  - $p[e/d] \xrightarrow{m} \surd \wedge r' = q$, which means that $\sum_{d:D} p \xrightarrow{m} \surd$ and $(\sum_{d:D} p) \mathbin{\|\!\_} q \xrightarrow{m} q$, with rRq, or
  - $p[e/d] \xrightarrow{m} p' \wedge r' = p' \parallel q$, which means that $\sum_{d:D} p \xrightarrow{m} p'$ and $(\sum_{d:D} p) \mathbin{\|\!\_} q \xrightarrow{m} p' \parallel q$, with $rRp' \parallel q$

A.18 *SUM7* $(\sum_{d:D} p)|y \doteq \sum_{d:D}(p|y)$

Note that q cannot contain d as it is a substitution for y.

- $(\sum_{d:D} p)|q \xrightarrow{m} \surd$, which means that $\sum_{d:D} p \xrightarrow{m'} \surd \wedge q \xrightarrow{n} \surd \wedge m = m' \oplus n$ and $\exists_{e:D}(p[e/d] \xrightarrow{m'} \surd)$ and $p[e/d]|q[e/d] \xrightarrow{m} \surd$ and $\sum_{d:D}(p|q) \xrightarrow{m} \surd$
- $\sum_{d:D}(p|q) \xrightarrow{m} \surd$, which means that $\exists_{e:D}(p[e/d]|q[e/d] \xrightarrow{m} \surd)$ and $p[e/d] \xrightarrow{m'} \surd \wedge q \xrightarrow{n} \surd \wedge m = m' \oplus n$ and $\sum_{d:D} p \xrightarrow{m'} \surd$ and $(\sum_{d:D} p)|q \xrightarrow{m} \surd$
- $(\sum_{d:D} p)|q \xrightarrow{m} r$, which means that
  - $\sum_{d:D} p \xrightarrow{m'} \surd \wedge q \xrightarrow{n} q' \wedge m = m' \oplus n \wedge r = q'$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m'} \surd)$ and $p[e/d]|q[e/d] \xrightarrow{m} q'$ and $\sum_{d:D}(p|q) \xrightarrow{m} q'$, with rRq', or

  - $\sum_{d:D} p \xrightarrow{m'} p' \wedge q \xrightarrow{n} \surd \wedge m = m' \oplus n \wedge r = p'$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m'} p')$ and $p[e/d]|q[e/d] \xrightarrow{m} p'$ and $\sum_{d:D}(p|q) \xrightarrow{m} p'$, with rRp', or
  - $\sum_{d:D} p \xrightarrow{m'} p' \wedge q \xrightarrow{n} q' \wedge m = m' \oplus n \wedge r = p' \parallel q'$, which means that $\exists_{e:D}(p[e/d] \xrightarrow{m'} p')$ and $p[e/d]|q[e/d] \xrightarrow{m} p' \parallel q'$ and $\sum_{d:D}(p|q) \xrightarrow{m} p' \parallel q'$, with $rRp' \parallel q'$

- $\sum_{d:D}(p|q) \xrightarrow{m} r$, which means that $\exists_{e:D}(p[e/d]|q[e/d] \xrightarrow{m} r') \wedge r = r'$ and
  - $p[e/d] \xrightarrow{m'} \surd \wedge q \xrightarrow{n} q' \wedge m = m' \oplus n \wedge r' = q'$, which means that $\sum_{d:D} p \xrightarrow{m} \surd$ and $(\sum_{d:D} p)|q \xrightarrow{m} q'$, with rRq', or
  - $p[e/d] \xrightarrow{m'} p' \wedge q \xrightarrow{n} \surd \wedge m = m' \oplus n \wedge r' = p'$, which means that $\sum_{d:D} p \xrightarrow{m} p'$ and $(\sum_{d:D} p)|q \xrightarrow{m} p'$, with rRp', or
  - $p[e/d] \xrightarrow{m'} p' \wedge q \xrightarrow{n} q' \wedge m = m' \oplus n \wedge r' = p' \parallel q'$, which means that $\sum_{d:D} p \xrightarrow{m} p'$ and $(\sum_{d:D} p)|q \xrightarrow{m} p' \parallel q'$, with $rRp' \parallel q'$

A.19 *SUM7'* $x|(\sum_{d:D} q) \doteq \sum_{d:D}(x|q)$

This proof is symmetrical to the proof of axiom *SUM*7.

A.20 *CM*1 $x \parallel y \doteq x \llfloor y + y \llfloor x + x|y$

We define E as $p \parallel qRq \parallel p$ (for all $p, q \in T_{pc}$).

- $p \parallel q \xrightarrow{m} \surd$, which means that $p \xrightarrow{n} \surd \wedge q \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $p|q \xrightarrow{m} \surd$ and $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} \surd$
- $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} \surd$, which means that
  - $p \llfloor q \xrightarrow{m} \surd$, which is not possible, or
  - $q \llfloor p \xrightarrow{m} \surd$, which is not possible, or
  - $p|q \xrightarrow{m} \surd$, which means that $p \xrightarrow{n} \surd \wedge q \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $p \parallel q \xrightarrow{m} \surd$
- $p \parallel q \xrightarrow{m} r$, which means that
  - $p \xrightarrow{m} \surd \wedge r = q$, which means that $p \llfloor q \xrightarrow{m} q$ and $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} q$, with rRq, or
  - $q \xrightarrow{m} \surd \wedge r = p$, which means that $q \llfloor p \xrightarrow{m} p$ and $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} p$, with rRp, or
  - $p \xrightarrow{m} p' \wedge r = p' \parallel q$, which means that $p \llfloor q \xrightarrow{m} p' \parallel q$ and $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} p' \parallel q$, with $rRp' \parallel q$, or
  - $q \xrightarrow{m} q' \wedge r = p \parallel q'$, which means that $q \llfloor p \xrightarrow{m} q' \parallel p$ and $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} q' \parallel p$, with $rRq' \parallel p$
- $p \llfloor q + q \llfloor p + p|q \xrightarrow{m} r$, which means that
  - $p \llfloor q \xrightarrow{m} r$, which means that
    - $p \xrightarrow{m} \surd \wedge r = q$, which means that $p \parallel q \xrightarrow{m} q$, with rRq, or
    - $p \xrightarrow{m} p' \wedge r = p' \parallel q$, which means that $p \parallel q \xrightarrow{m} p' \parallel q$, with $rRp' \parallel q$, or
  - $q \llfloor p \xrightarrow{m} r$, which means that

  - $q \xrightarrow{m} \surd \wedge r = p$, which means that $p \parallel q \xrightarrow{m} p$, with rRp, or
  - $q \xrightarrow{m} q' \wedge r = q' \parallel p$, which means that $p \parallel q \xrightarrow{m} p \parallel q'$, with $rRp \parallel q'$, or
- $p|q \xrightarrow{m} r$, which means that
  - $p \xrightarrow{n} p' \wedge q \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r = p' \parallel q'$, which means that $p \parallel q \xrightarrow{m} p' \parallel q'$, with $rRp' \parallel q'$, or
  - $p \xrightarrow{n} \surd \wedge q \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r = q'$, which means that $p \parallel q \xrightarrow{m} q'$, with rRq', or
  - $p \xrightarrow{n} p' \wedge q \xrightarrow{n'} \surd \wedge m = n \oplus n' \wedge r = p'$, which means that $p \parallel q \xrightarrow{m} p'$, with rRp'

- $p \parallel q \xrightarrow{m} \surd$, which means that $p \xrightarrow{n} \surd \wedge q \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $q \parallel p \xrightarrow{m} \surd$
- $q \parallel p \xrightarrow{m} \surd$, which is symmetrical to the proof above
- $p \parallel q \xrightarrow{m} r$, which means that
  - $p \xrightarrow{m} \surd \wedge r = q$, which means that $q \parallel p \xrightarrow{m} q$, with rRq, or
  - $q \xrightarrow{m} \surd \wedge r = p$, which means that $q \parallel p \xrightarrow{m} p$, with rRp, or
  - $p \xrightarrow{m} p' \wedge r = p' \parallel q$, which means that $q \parallel p \xrightarrow{m} q \parallel p'$, with $rRq \parallel p'$, or
  - $q \xrightarrow{m} q' \wedge r = p \parallel q'$, which means that $q \parallel p \xrightarrow{m} q' \parallel p$, with $rRq' \parallel p$, or
  - $p \xrightarrow{n} \surd \wedge q \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r = q'$, which means that $q \parallel p \xrightarrow{m} q'$, with rRq', or
  - $p \xrightarrow{n} p' \wedge q \xrightarrow{n'} \surd \wedge m = n \oplus n' \wedge r = p'$, which means that $q \parallel p \xrightarrow{m} p'$, with rRp', or
  - $p \xrightarrow{n} p' \wedge q \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r = p' \parallel q'$, which means that $q \parallel p \xrightarrow{m} q' \parallel p'$, with $rRq' \parallel p'$, or
- $q \parallel p \xrightarrow{m} r$, which is symmetrical to the proof above

A.21 $CM2$ $\alpha \mathbin{\|\!\_} x \doteq \alpha{\cdot}x$

- Both $\alpha \mathbin{\|\!\_} p$ and $\alpha{\cdot}p$ can not terminate
- $\alpha \mathbin{\|\!\_} p \xrightarrow{m} p'$, which means that $\alpha \xrightarrow{m} \surd \wedge p' = p$ and $\alpha{\cdot}p \xrightarrow{m} p$, with $p'Rp$
- $\alpha{\cdot}p \xrightarrow{m} p'$, which means that $\alpha \xrightarrow{m} \surd \wedge p' = p$ and $\alpha \mathbin{\|\!\_} p \xrightarrow{m} p$, with $p'Rp$

A.22 $CM3$ $\alpha{\cdot}x \mathbin{\|\!\_} y \doteq \alpha{\cdot}(x \parallel y)$

- Both $\alpha{\cdot}p \mathbin{\|\!\_} q$ and $\alpha{\cdot}(p \parallel q)$ can not terminate
- $\alpha{\cdot}p \mathbin{\|\!\_} q \xrightarrow{m} q'$, which means that $\alpha{\cdot}p \xrightarrow{m} p' \wedge q' = p' \parallel q$ and $\alpha \xrightarrow{m} \surd \wedge p' = p$ and $\alpha{\cdot}(p \parallel q) \xrightarrow{m} p \parallel q$, with $q'Rp \parallel q$
- $\alpha{\cdot}(p \parallel q) \xrightarrow{m} q'$, which means that $\alpha \xrightarrow{m} \surd \wedge q' = p \parallel q$ and $\alpha{\cdot}p \xrightarrow{m} p$ and $\alpha{\cdot}p \mathbin{\|\!\_} q \xrightarrow{m} p \parallel q$, with $q'Rp \parallel q$

A.23 $CM4$ $(x+y) \mathbin{\|\!\_} z \doteq x \mathbin{\|\!\_} z + y \mathbin{\|\!\_} z$

- $(p+q) \mathbin{\|\!\_} r$ can not terminate
- $p \mathbin{\|\!\_} r + q \mathbin{\|\!\_} r$ can not terminate as
  - $p \mathbin{\|\!\_} r$ can not terminate and
  - $q \mathbin{\|\!\_} r$ can not terminate
- $(p+q) \mathbin{\|\!\_} r \xrightarrow{m} r'$, which means that
  - $p+q \xrightarrow{m} \surd \wedge r' = r$, which means that
    - $p \xrightarrow{m} \surd$, which means that $p \mathbin{\|\!\_} r \xrightarrow{m} r$ and $p \mathbin{\|\!\_} r + q \mathbin{\|\!\_} r \xrightarrow{m} r$, with $r'Rr$, or
    - $q \xrightarrow{m} \surd$, which means that $q \mathbin{\|\!\_} r \xrightarrow{m} r$ and $p \mathbin{\|\!\_} r + q \mathbin{\|\!\_} r \xrightarrow{m} r$, with $r'Rr$, or
  - $p+q \xrightarrow{m} q' \wedge r' = q' \parallel r$, which means that
    - $p \xrightarrow{m} q'$, which means that $p \mathbin{\|\!\_} r \xrightarrow{m} q' \parallel r$ and $p \mathbin{\|\!\_} r + q \mathbin{\|\!\_} r \xrightarrow{m} q' \parallel r$, with $r'Rq' \parallel r$, or
    - $q \xrightarrow{m} q'$, which means that $q \mathbin{\|\!\_} r \xrightarrow{m} q' \parallel r$ and $p \mathbin{\|\!\_} r + q \mathbin{\|\!\_} r \xrightarrow{m} q' \parallel r$, with $r'Rq' \parallel r$
- $p \mathbin{\|\!\_} r + q \mathbin{\|\!\_} r \xrightarrow{m} r'$, which means that
  - $p \mathbin{\|\!\_} r \xrightarrow{m} r'$, which means that
    - $p \xrightarrow{m} \surd \wedge r' = r$, which means that $p+q \xrightarrow{m} \surd$ and $(p+q) \mathbin{\|\!\_} r \xrightarrow{m} r$, with $r'Rr$, or
    - $p \xrightarrow{m} p' \wedge r' = p' \parallel r$, which means that $p+q \xrightarrow{m} p'$ and $(p+q) \mathbin{\|\!\_} r \xrightarrow{m} p' \parallel r$, with $r'Rp' \parallel r$, or
  - $q \mathbin{\|\!\_} r \xrightarrow{m} r'$, which means that
    - $q \xrightarrow{m} \surd \wedge r' = r$, which means that $p+q \xrightarrow{m} \surd$ and $(p+q) \mathbin{\|\!\_} r \xrightarrow{m} r$, with $r'Rr$, or
    - $q \xrightarrow{m} q' \wedge r' = q' \parallel r$, which means that $p+q \xrightarrow{m} q'$ and $(p+q) \mathbin{\|\!\_} r \xrightarrow{m} q' \parallel r$, with $r'Rq' \parallel r$

A.24 $CM5$ $\alpha{\cdot}x|\beta \doteq (\alpha|\beta)\cdot x$

- Both $\alpha{\cdot}p|\beta$ and $(\alpha|\beta)\cdot p$ cannot terminate
- $\alpha{\cdot}p|\beta \xrightarrow{m} p'$, which means that $\alpha{\cdot}p \xrightarrow{n} p' \wedge \beta \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $\alpha \xrightarrow{n} \surd \wedge p' = p$ and $\alpha|\beta \xrightarrow{m} \surd$ and $(\alpha|\beta)\cdot p \xrightarrow{m} p$, with $p'Rp$
- $(\alpha|\beta)\cdot p \xrightarrow{m} p'$, which means that $\alpha|\beta \xrightarrow{m} \surd \wedge p' = p$ and $\alpha \xrightarrow{n} \surd \wedge \beta \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $\alpha{\cdot}p \xrightarrow{n} p$ and $\alpha{\cdot}p|\beta \xrightarrow{m} p$, with $p'Rp$

A.25 $CM6$ $\alpha|\beta{\cdot}x \doteq (\alpha|\beta)\cdot x$

Proof is symmetrical to the proof of axiom $CM5$.

A.26 $CM7$ $\alpha{\cdot}x|\beta{\cdot}y \doteq (\alpha|\beta)\cdot(x \parallel y)$

- Both $\alpha{\cdot}p|\beta{\cdot}q$ and $(\alpha|\beta)\cdot(p \parallel q)$ cannot terminate
- $\alpha{\cdot}p|\beta{\cdot}q \xrightarrow{m} r$, which means that $\alpha{\cdot}p \xrightarrow{n} p' \wedge \beta{\cdot}q \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r = p' \parallel q'$ and $\alpha \xrightarrow{n} \surd \wedge p' = p$ and $\beta \xrightarrow{n'} \surd \wedge q' = q$ and $\alpha|\beta \xrightarrow{m} \surd$ and $(\alpha|\beta)\cdot(p \parallel q) \xrightarrow{m} p \parallel q$, with $rRp \parallel q$
- $(\alpha|\beta)\cdot(p \parallel q) \xrightarrow{m} r$, which means that $\alpha|\beta \xrightarrow{m} \surd \wedge r = p \parallel q$ and $\alpha \xrightarrow{n} \surd \wedge \beta \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $\alpha{\cdot}p \xrightarrow{n} p$ and $\beta{\cdot}q \xrightarrow{n'} q$ and $\alpha{\cdot}p|\beta{\cdot}q \xrightarrow{m} p \parallel q$, with $rRp \parallel q$

A.27 *CM*8 $(x+y)|z \doteq x|z+y|z$

- $(p+q)|r \xrightarrow{m} \surd$, which means $p+q \xrightarrow{n} \surd \wedge r \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and
  - $p \xrightarrow{n} \surd$, which means $p|r \xrightarrow{m} \surd$ and $p|r+q|r \xrightarrow{m} \surd$, or
  - $q \xrightarrow{n} \surd$, which means $q|r \xrightarrow{m} \surd$ and $p|r+q|r \xrightarrow{m} \surd$
- $p|r+q|r \xrightarrow{m} \surd$, which means that
  - $p|r \xrightarrow{m} \surd$, which means $p \xrightarrow{n} \surd \wedge r \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $p+q \xrightarrow{n} \surd$ and $(p+q)|r \xrightarrow{m} \surd$, or
  - $q|r \xrightarrow{m} \surd$, which means $q \xrightarrow{n} \surd \wedge r \xrightarrow{n'} \surd \wedge m = n \oplus n'$ and $p+q \xrightarrow{n} \surd$ and $(p+q)|r \xrightarrow{m} \surd$
- $(p+q)|r \xrightarrow{m} r'$, which means that
  - $p+q \xrightarrow{n} \surd \wedge r \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r' = q'$, which means that
    - $p \xrightarrow{n} \surd$, which means that $p|r \xrightarrow{m} q'$ and $p|r+q|r \xrightarrow{m} q'$, which $r'Rq'$, or
    - $q \xrightarrow{n} \surd$, which means that $q|r \xrightarrow{m} q'$ and $p|r+q|r \xrightarrow{m} q'$, which $r'Rq'$, or
  - $p+q \xrightarrow{n} q' \wedge r \xrightarrow{n'} \surd \wedge m = n \oplus n' \wedge r' = q'$, which means that
    - $p \xrightarrow{n} p' \wedge q' = p'$, which means that $p|r \xrightarrow{m} p'$ and $p|r+q|r \xrightarrow{m} p'$, which $r'Rp'$, or
    - $q \xrightarrow{n} p' \wedge q' = p'$, which means that $q|r \xrightarrow{m} p'$ and $p|r+q|r \xrightarrow{m} p'$, which $r'Rp'$, or
  - $p+q \xrightarrow{n} p' \wedge r \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge r' = p' \parallel q'$, which means that
    - $p \xrightarrow{n} p'' \wedge p' = p''$, which means that $p|r \xrightarrow{m} p'' \parallel r'$ and $p|r+q|r \xrightarrow{m} p' \parallel r'$, which $r'Rp'' \parallel q'$, or
    - $q \xrightarrow{n} p'' \wedge p' = p''$, which means that $q|r \xrightarrow{m} p'' \parallel r'$ and $p|r+q|r \xrightarrow{m} p' \parallel r'$, which $r'Rp'' \parallel q'$
- $p|r+q|r \xrightarrow{m} r'$, which means that
  - $p|r \xrightarrow{m} p' \wedge r' = p'$, which means that
    - $p \xrightarrow{n} \surd \wedge r \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge p' = q'$, which means $p+q \xrightarrow{n} \surd$ and $(p+q)|r \xrightarrow{m} q'$, with $r'Rq'$, or
    - $p \xrightarrow{n} q' \wedge r \xrightarrow{n'} \surd \wedge m = n \oplus n' \wedge p' = q'$, which means $p+q \xrightarrow{n} q'$ and $(p+q)|r \xrightarrow{m} q'$, with $r'Rq'$, or
    - $p \xrightarrow{n} p'' \wedge r \xrightarrow{n'} q' \wedge m = n \oplus n' \wedge p' = p'' \parallel q'$, which means $p+q \xrightarrow{n} p''$ and $(p+q)|r \xrightarrow{m} p'' \parallel q'$, with $r'Rp'' \parallel q'$
  - $q|r \xrightarrow{m} p' \wedge r' = p'$, which is symmetrical to the previous

A.28 *CM*9 $x|(y+z) \doteq x|y+x|z$

Proof is symmetrical to the proof of *CM*8.

A.29 *CM*10 $\langle\vec{a}\rangle|\langle\vec{b}\rangle \doteq \langle\vec{a}, \vec{b}\rangle$

- $\langle\vec{a}\rangle|\langle\vec{b}\rangle \xrightarrow{m} \surd \wedge m = [\![\langle\vec{a}\rangle]\!] \oplus [\![\langle\vec{b}\rangle]\!] = [\![\langle\vec{a}, \vec{b}\rangle]\!]$ and $\langle\vec{a}, \vec{b}\rangle \xrightarrow{m} \surd$.
- Both $\langle\vec{a}\rangle|\langle\vec{b}\rangle$ and $\langle\vec{a}, \vec{b}\rangle$ cannot do any transition.

A.30 $CD1$ $\delta|\alpha \doteq \delta$

Both $\delta|\alpha$ and δ cannot terminate or make any transition.

A.31 $CD2$ $\alpha|\delta \doteq \delta$

Both $\alpha|\delta$ and δ cannot terminate or make any transition.

A.32 VD $\nabla_V(\delta) \doteq \delta$

Both $\nabla_V(\delta)$ and δ cannot terminate or make any transition.

A.33 $V1$ $\nabla_V(\alpha) \doteq \alpha$ *if* $\mu([\![\alpha]\!]) \in [\![V]\!] \cup \{[]\}$

Assuming $\mu([\![\alpha]\!]) \in [\![V]\!] \cup \{[]\}$:

- $\nabla_V(\alpha)\xrightarrow{m}\surd$, which means that $\alpha\xrightarrow{m}\surd$
- $\alpha\xrightarrow{m}\surd$, which means that $\nabla_V(\alpha)\xrightarrow{m}\surd$

Both $\nabla_V(\alpha)$ and α cannot do any transition.

A.34 $V2$ $\nabla_V(\alpha) \doteq \delta$ *if* $\mu([\![\alpha]\!]) \notin [\![V]\!] \cup \{[]\}$

Assuming $\mu([\![\alpha]\!]) \notin [\![V]\!] \cup \{[]\}$, both $\nabla_V(\alpha)$ and δ cannot terminate or make any transition.

A.35 $V3$ $\nabla_V(x+y) \doteq \nabla_V(x) + \nabla_V(y)$

- $\nabla_V(p+q)\xrightarrow{m}\surd$, which means that $p+q\xrightarrow{m}\surd \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and
  - $p\xrightarrow{m}\surd$, which means that $\nabla_V(p)\xrightarrow{m}\surd$ and $\nabla_V(p)+\nabla_V(q)\xrightarrow{m}\surd$, or
  - $q\xrightarrow{m}\surd$, which means that $\nabla_V(q)\xrightarrow{m}\surd$ and $\nabla_V(p)+\nabla_V(q)\xrightarrow{m}\surd$
- $\nabla_V(p)+\nabla_V(q)\xrightarrow{m}\surd$, which means that
  - $\nabla_V(p)\xrightarrow{m}\surd$, which means that $p\xrightarrow{m}\surd \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $p+q\xrightarrow{m}\surd$ and $\nabla_V(p+q)\xrightarrow{m}\surd$, or
  - $\nabla_V(q)\xrightarrow{m}\surd$, which means that $q\xrightarrow{m}\surd \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $p+q\xrightarrow{m}\surd$ and $\nabla_V(p+q)\xrightarrow{m}\surd$
- $\nabla_V(p+q)\xrightarrow{m}r$, which means that $p+q\xrightarrow{m}r' \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \wedge r = z'$ and
  - $p\xrightarrow{m}p' \wedge r' = p'$, which means that $\nabla_V(p)\xrightarrow{m}p'$ and $\nabla_V(p)+\nabla_V(q)\xrightarrow{m}p'$, with rRp', or
  - $q\xrightarrow{m}q' \wedge r' = q'$, which means that $\nabla_V(q)\xrightarrow{m}q'$ and $\nabla_V(p)+\nabla_V(q)\xrightarrow{m}q'$, with rRq'
- $\nabla_V(p)+\nabla_V(q)\xrightarrow{m}r$, which means that
  - $\nabla_V(p)\xrightarrow{m}p' \wedge r = p'$, which means that $p\xrightarrow{m}p'' \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \wedge p' = p''$ and $p+q\xrightarrow{m}p''$ and $\nabla_V(p+q)\xrightarrow{m}p''$, with rRp'', or
  - $\nabla_V(q)\xrightarrow{m}q' \wedge r = q'$, which means that $q\xrightarrow{m}q'' \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \wedge q' = q''$ and $p+q\xrightarrow{m}q''$ and $\nabla_V(p+q)\xrightarrow{m}q''$, with rRq''

A.36 $V4$ $\nabla_V(x \cdot y) \doteq \nabla_V(x) \cdot \nabla_V(y)$

Both $\nabla_V(p \cdot q)$ and $\nabla_V(p) \cdot \nabla_V(q)$ cannot terminate.

- $\nabla_V(p \cdot q \xrightarrow{m} r)$, which means that $p \cdot q \xrightarrow{m} r' \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \wedge r = \nabla_V(r')$ and
  - $p \xrightarrow{m} \checkmark \wedge r' = q$, which means that $\nabla_V(p) \xrightarrow{m} \checkmark$ and $\nabla_V(p) \cdot \nabla_V(q) \xrightarrow{m} \nabla_V(q)$, with $rR\nabla_V(q)$, or
  - $p \xrightarrow{m} p' \wedge r' = p' \cdot q$, which means that $\nabla_V(p) \xrightarrow{m} \nabla_V(p')$ and $\nabla_V(p) \cdot \nabla_V(q) \xrightarrow{m} \nabla_V(p') \cdot \nabla_V(q)$, with $rR\nabla_V(p' \cdot q)$
- $\nabla_V(p) \cdot \nabla_V(q) \xrightarrow{m} r$, which means that
  - $\nabla_V(p) \xrightarrow{m} \checkmark \wedge r = \nabla_V(q)$, which means that $p \xrightarrow{m} \checkmark \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $p \cdot q \xrightarrow{m} q$ and $\nabla_V(p \cdot q) \xrightarrow{m} \nabla_V(q)$, with $rR\nabla_V(q)$, or
  - $\nabla_V(p) \xrightarrow{m} p' \wedge r = p' \cdot \nabla_V(q)$, which means that $p \xrightarrow{m} p'' \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \wedge p' = \nabla_V(p'')$ and $p \cdot q \xrightarrow{m} p'' \cdot q$ and $\nabla_V(p \cdot q) \xrightarrow{m} \nabla_V(p'' \cdot q)$, with $rR\nabla_V(p'' \cdot q)$

A.37 $V6$ $\nabla_V(\sum_{d:D} p) \doteq \sum_{d:D}(\nabla_V(p))$

- $\nabla_V(\sum_{d:D} p) \xrightarrow{m} \checkmark$, which means that $\sum_{d:D} p \xrightarrow{m} \checkmark \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $\exists_{e:D}(p[e/d] \xrightarrow{m} \checkmark)$ and $\nabla_V(p[e/d]) \xrightarrow{m} \checkmark$ and $\sum_{d:D}(\nabla_V(p)) \xrightarrow{m} \checkmark$
- $\sum_{d:D}(\nabla_V(p)) \xrightarrow{m} \checkmark$, which means that $\exists_{e:D}(\nabla_V(p[e/d]) \xrightarrow{m} \checkmark)$ and $p[e/d] \xrightarrow{m} \checkmark \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $\sum_{d:D} p \xrightarrow{m} \checkmark$ and $\nabla_V(\sum_{d:D} p) \xrightarrow{m} \checkmark$
- $\nabla_V(\sum_{d:D} p) \xrightarrow{m} p'$, which means that $\sum_{d:D} p \xrightarrow{m} p' \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $\exists_{e:D}(p[e/d] \xrightarrow{m} p')$ and $\nabla_V(p[e/d]) \xrightarrow{m} p'$ and $\sum_{d:D}(\nabla_V(p)) \xrightarrow{m} p'$, with $p'Rp'$
- $\sum_{d:D}(\nabla_V(p)) \xrightarrow{m} p'$, which means that $\exists_{e:D}(\nabla_V(p[e/d]) \xrightarrow{m} p')$ and $p[e/d] \xrightarrow{m} p' \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $\sum_{d:D} p \xrightarrow{m} p'$ and $\nabla_V(\sum_{d:D} p) \xrightarrow{m} p'$, with $p'Rp'$

A.38 DD $\partial_H(\delta) \doteq \delta$

Both $\partial_H(\delta)$ and δ cannot terminate or make any transition.

A.39 $D1$ $\partial_H(\alpha) \doteq \alpha$ *if* $\mu([\![\alpha]\!]) \cap [\![H]\!] = \emptyset$

Both $\partial_H(\alpha)$ and α can only terminate (with $\xrightarrow{[\![\alpha]\!]}$).

A.40 $D2$ $\partial_H(\alpha) \doteq \delta$ *if* $\mu([\![\alpha]\!]) \cap [\![H]\!] \neq \emptyset$

Both $\partial_H(\alpha)$ and δ cannot terminate or make any transition.

A.41 $D3$ $\partial_H(x + y) \doteq \partial_H(x) + \partial_H(y)$

Proof is similar to the proof of axiom $V3$.

A.42 $D4$ $\partial_H(x \cdot y) \doteq \partial_H(x) \cdot \partial_H(y)$

Proof is similar to the proof of axiom $V4$.

A.43 *D6* $\partial_H(\sum_{d:D} p) \doteq \sum_{d:D}(\partial_H(p))$

Proof is similar to the proof of axiom $V6$.

A.44 *TID* $\tau_I(\delta) \doteq \delta$

Both $\tau_I(\delta)$ and δ cannot terminate or make any transition.

A.45 *TI1* $\tau_I(\alpha) \doteq \beta$ *with* $[\![\beta]\!] = \theta([\![\alpha]\!], I)$

Both $\tau_I(\alpha)$ and β can only terminate (with $\xrightarrow{\theta([\![\alpha]\!],I)}$).

A.46 *TI3* $\tau_I(x+y) \doteq \tau_I(x) + \tau_I(y)$

Proof is similar to the proof of axiom $V3$.

A.47 *TI4* $\tau_I(x \cdot y) \doteq \tau_I(x) \cdot \tau_I(y)$

Proof is similar to the proof of axiom $V4$.

A.48 *TI6* $\tau_I(\sum_{d:D} p) \doteq \sum_{d:D}(\tau_I(p))$

Proof is similar to the proof of axiom $V6$.

A.49 *RD* $\rho_R(\delta) \doteq \delta$

Both $\rho_R(\delta)$ and δ cannot terminate or make any transition.

A.50 *R1* $\rho_R(\alpha) \doteq \beta$ *with* $[\![\beta]\!] = [\![R]\!] \bullet [\![\alpha]\!]$

Both $\rho_R(\alpha)$ and β can only terminate (with $\xrightarrow{[\![R]\!]\bullet[\![\alpha]\!]}$).

A.51 *R3* $\rho_R(x+y) \doteq \rho_R(x) + \rho_R(y)$

Proof is similar to the proof of axiom $V3$.

A.52 *R4* $\rho_R(x \cdot y) \doteq \rho_R(x) \cdot \rho_R(y)$

Proof is similar to the proof of axiom $V4$.

A.53 *R6* $\rho_R(\sum_{d:D} p) \doteq \sum_{d:D}(\rho_R(p))$

Proof is similar to the proof of axiom $V6$.

A.54 *GD* $\Gamma_C(\delta) \doteq \delta$

Both $\rho_R(\delta)$ and δ cannot terminate or make any transition.

A.55 *G1* $\Gamma_C(\alpha) \doteq \beta$ *with* $[\![\beta]\!] = \gamma([\![\alpha]\!], [\![C]\!])$

Process $\Gamma_C(\alpha)$ can only terminate with $\gamma([\![\alpha]\!], [\![C]\!])$ and β only with $[\![\beta]\!]$, which are equal according to the condition of the axiom.

A.56 $G3$ $\Gamma_C(x+y) \doteq \Gamma_C(x) + \Gamma_C(y)$

Proof is similar to the proof of axiom $V3$.

A.57 $G4$ $\Gamma_C(x \cdot y) \doteq \Gamma_C(x) \cdot \Gamma_C(y)$

Proof is similar to the proof of axiom $V4$.

A.58 $G6$ $\Gamma_C(\sum_{d:D} p) \doteq \sum_{d:D}(\Gamma_C(p))$

Proof is similar to the proof of axiom $V6$.

B Soundness proofs of τ axioms

The structure of the following proofs is the same as before, except for the fact that we prove R to be a branching bisimulation and *rooted*(t,u) (for axiom $t \doteq u$).

B.1 $T1$ $x \cdot \tau \doteq x$

We define E as $\tau R \surd$.

- $p \cdot \tau R p$
    - $p \cdot \tau \downarrow$, which is not possible
    - $p \downarrow$, which means that $p = \surd$, which is not possible
    - $p \cdot \tau \xrightarrow{m} p'$, which means that
        - $p \xrightarrow{m} \surd \wedge p' = \tau$, with $p' R \surd$, or
        - $p \xrightarrow{m} p'' \wedge p' = p'' \cdot \tau$, with $p' R p''$, or
    - $p \xrightarrow{m} p'$, which means that $p \cdot \tau \xrightarrow{m} p' \cdot \tau$, with $p' R p' \cdot \tau$
- *rooted*$(p \cdot \tau, p)$, which is similar to the previous case
- $\tau R \surd$
    - $\tau \downarrow$, which is not possible
    - $\surd \downarrow$, which means that $\tau \xrightarrow{[]} \surd \downarrow$
    - $\tau \xrightarrow{m} p$, which means that $p = \surd \wedge m = []$, with $p R \surd$
    - $\surd \xrightarrow{m} p$, which is not possible

B.2 $T2$ $x \cdot (\tau \cdot (y+z) + y) = x \cdot (y+z)$

We define E as $\tau \cdot (q+r) + q R q + r$ for all $q, r \in T_{pc}$.

- $p \cdot (\tau \cdot (q+r) + q) R p \cdot (q+r)$
    - $p \cdot (\tau \cdot (q+r) + q) \downarrow$, which is not possible
    - $p \cdot (q+r) \downarrow$, which is not possible
    - $p \cdot (\tau \cdot (q+r) + q) \xrightarrow{m} r'$, which means that
        - $p \xrightarrow{m} \surd \wedge r' = \tau \cdot (q+r) + q$, which means that $p \cdot (q+r) \xrightarrow{m} q + r$, with $r' R q + r$, or

- $p \xrightarrow{m} p' \wedge r' = p' \cdot (\tau \cdot (q+r) + q)$, which means that $p \cdot (q+r) \xrightarrow{m} p' \cdot (q+r)$, with $r' R p' \cdot (q+r)$, or
- $p \cdot (q+r) \xrightarrow{m} r'$, which means that
  - $p \xrightarrow{m} \checkmark \wedge r' = q + r$, which means that $p \cdot (\tau \cdot (q+r) + q) \xrightarrow{m} \tau \cdot (q+r) + q$, with $r' R \tau \cdot (q+r) + q$, or
  - $p \xrightarrow{m} p' \wedge r' = p' \cdot (q+r)$, which means that $p \cdot (\tau \cdot (q+r) + q) \xrightarrow{m} p' \cdot (\tau \cdot (q+r) + q)$, with $r' R p' \cdot (\tau \cdot (q+r) + q)$, or

- $\mathit{rooted}(p \cdot (\tau \cdot (q+r) + q), p \cdot (q+r))$, which is similar to the previous case
- $\tau \cdot (q+r) + q R q + r$
  - $\tau \cdot (q+r) + q \downarrow$, which is not possible
  - $q + r \downarrow$, which is not possible
  - $\tau \cdot (q+r) + q \xrightarrow{m} p$, which means that
    - $\tau \cdot (q+r) \xrightarrow{m} p$, which means that
      - $\tau \xrightarrow{m} \checkmark \wedge p = q + r$, which means that $m = []$ and $pRq + r$, or
      - $\tau \xrightarrow{m} p'$, which is not possible, or
    - $q \xrightarrow{m} p$, which means that $q + r \xrightarrow{m} p$, with pRx
  - $q + r \xrightarrow{m} p$, which means that $\tau \cdot (q+r) \xrightarrow{[]} q + r \xrightarrow{m} p$, with pRp

C Soundness proofs of alphabet axioms

To prove the soundness of these alphabet axioms, we use a different approach as before, except for the axioms containing a parallel operator. As the operators used in these axioms are all defined in a similar way, namely:

$$\frac{x \xrightarrow{m} \checkmark}{O(x) \xrightarrow{f(m)} \checkmark}\, C(m) \qquad \frac{x \xrightarrow{m} x'}{O(x) \xrightarrow{f(m)} O(x')}\, C(m)$$

If we now wish to prove an axiom $O_1(\ldots O_n(x)\ldots) \doteq O_{n+1}(\ldots O_{n'}(x)\ldots)$, we get the following:

- $O_1(\ldots O_n(x)\ldots) \xrightarrow{m} \checkmark$, which means that $O_2(\ldots O_n(x)\ldots) \xrightarrow{m_1} \checkmark \wedge C_1(m_1) \wedge m = f_1(m_1) \ldots x \xrightarrow{m_n} \checkmark \wedge C_n(m_n) \wedge m_{n-1} = f_n(m_n)$ and therefore $C_{n+1}(m_n) \wedge \ldots \wedge C_{n'}(f_{n'-1}(\ldots f_{n+1}(m_n)\ldots))$ and $m = f'_n(\ldots f_{n+1}(m_n)\ldots)$ and $O_{n+1}(\ldots O_{n'}(x)\ldots) \xrightarrow{m} \checkmark$
- Etc.

So what really has to be proven is that for each m, such that $x \xrightarrow{m} \checkmark \vee x \xrightarrow{m} x'$, it holds that $C_n(m) \wedge \ldots \wedge C_1(f_2(\ldots f_n(m))) \equiv C_{n'}(m) \wedge \ldots \wedge C_{n+1}(f_{n+2}(\ldots f_{n'}(m)))$ and $C_n(m) \wedge \ldots \wedge C_1(f_2(\ldots f_n(m))) \Rightarrow f_1(\ldots f_n(m)) = f_{n+1}(\ldots f_{n'}(m))$.

Lemma C.1. Let $v \in \mathbb{B}(\mathcal{N}_A)$. Also, let $S \subseteq \mathcal{N}_A$. The following holds:

$$v \cap S = \mathcal{N}(\{v\}) \cap S$$

Proof C.1. Induction on the structure of v. Case $v = []$:

$$
\begin{array}{cl}
& [] \cap S \\
= & \quad \{ \text{ def. } \cap \ \} \\
& \emptyset \\
= & \quad \{ \text{ calculus } \} \\
& \emptyset \cap S \\
= & \quad \{ \text{ def. } \mathcal{N} \ \} \\
& \mathcal{N}(\{[]\}) \cap S
\end{array}
$$

Case $v = [a] \oplus w$:

$$
\begin{array}{cl}
& ([a] \oplus w) \cap S \\
= & \quad \{ \text{ def. } \cap \ \} \\
& w \cap S \qquad\qquad \textit{if } a \notin S \\
& \{a\} \cup (w \cap S) \qquad \textit{if } a \in S \\
= & \quad \{ \text{ induction hypothesis } \} \\
& \mathcal{N}(\{w\}) \cap S \qquad\qquad \textit{if } a \notin S \\
& \{a\} \cup (\mathcal{N}(\{w\}) \cap S) \qquad \textit{if } a \in S \\
= & \quad \{ \text{ calculus } \} \\
& (\{a\} \cup \mathcal{N}(\{w\})) \cap S \qquad\qquad \textit{if } a \notin S \\
& (\{a\} \cup \mathcal{N}(\{w\})) \cap (\{a\} \cup S) \qquad \textit{if } a \in S \\
= & \quad \{ \text{ calculus } \} \\
& (\{a\} \cup \mathcal{N}(\{w\})) \cap S \\
= & \quad \{ \text{ def. } \mathcal{N} \ \} \\
& \mathcal{N}(\{[a] \oplus w\}) \cap S
\end{array}
$$

□

Lemma C.2. Let $N_{\mathbb{B}} = \{n \mid n \in \mathbb{B}(\mathcal{N}_A) \wedge 1 \leq |n|\}$ and $C : N_{\mathbb{B}} \to (\mathcal{N}_A \cup \{\tau\})$. Also, let $\mathcal{N}(dom(C)) \cap rng(C) = \emptyset$ and $m, n \in \mathbb{B}(A)$. The following holds:

$$\gamma(m \oplus n, C) = \gamma(m \oplus \gamma(n, C), C)$$

Proof C.2. Induction on the length of n and by the cases of γ. Case $|n| = 0$:

$$
\begin{array}{cl}
& \gamma(m \oplus n, C) \\
= & \quad \{ \ n = [] \ \} \\
& \gamma(m \oplus [], C) \\
= & \quad \{ \text{ def. } \gamma \ \} \\
& \gamma(m \oplus \gamma([], C), C) \\
= & \quad \{ \ n = [] \ \} \\
& \gamma(m \oplus \gamma(n, C), C)
\end{array}
$$

Case $|n| > 0$ and $\exists_{n',o}(n = n' \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n') \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n', \vec{d}))))$:

$$
\begin{array}{cl}
& \gamma(m \oplus n, C) \\
= & \quad \{ \text{ take } n' \text{ and } o \text{ with } n = n' \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n') \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n', \vec{d})))) \ \} \\
& \gamma(m \oplus n' \oplus o, C) \\
= & \quad \{ \text{ take } \langle b,a \rangle \in C \text{ with } b = \mu(n') \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n', \vec{d})))), \text{ def. } \gamma \ \}
\end{array}
$$

$$
\begin{array}{ll}
 & [a(\vec{d})] \oplus \gamma(m \oplus o, C) \\
= & \quad \{ \text{ induction hypothesis } \} \\
 & [a(\vec{d})] \oplus \gamma(m \oplus \gamma(o, C), C) \\
= & \quad \{ \mathcal{N}(dom(C)) \cap rng(C) = \emptyset,\ a \in rng(C) \} \\
 & \gamma(m \oplus [a(\vec{d})] \oplus \gamma(o, C), C) \\
= & \quad \{ \text{ def. } \gamma \text{ and } a \} \\
 & \gamma(m \oplus \gamma(n' \oplus o, C), C) \\
= & \quad \{ \text{ def. } n' \text{ and } o \} \\
 & \gamma(m \oplus \gamma(n, C), C)
\end{array}
$$

Case $|n| > 0$ and $\exists_{n',o}(n = n' \oplus o \wedge \exists_{\langle b,\tau\rangle \in C}(b = \mu(n') \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n', \vec{d}))))$, which is similar to the previous case.
Case $|n| > 0$ and $\neg\exists_{n',o}(n = n' \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n') \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n', \vec{d}))))$:

$$
\begin{array}{ll}
 & \gamma(m \oplus n, C) \\
= & \quad \{ \neg\exists_{n',o}(n = n' \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n') \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n', \vec{d})))),\ \text{def. } \gamma \} \\
 & \gamma(m \oplus \gamma(n, C), C)
\end{array}
$$

□

C.1 *VA1* $\nabla_V(x) \doteq x$ *if* $\alpha_\nu(x) \subseteq [\![V]\!]$

$$
\begin{array}{ll}
 & \mu(m) \in [\![V]\!] \\
\equiv & \quad \{ \alpha_\nu(x) \subseteq [\![V]\!] \} \\
 & \mu(m) \in \alpha_\nu(x) \vee \mu(m) \in [\![V]\!] \\
\equiv & \quad \{ \text{ def. } \alpha_\nu \} \\
 & \mathit{true} \vee \mu(m) \in [\![V]\!] \\
\equiv & \quad \{ \text{ identity } \vee \} \\
 & \mathit{true}
\end{array}
$$

C.2 *VA2* $\nabla_V(x) \doteq \delta$ *if* $[\![V]\!] \cap \alpha_\nu(x) = \emptyset$

$$
\begin{array}{ll}
 & \mu(m) \in [\![V]\!] \\
\equiv & \quad \{ \text{ def. } \alpha_\nu \} \\
 & \mu(m) \in [\![V]\!] \wedge \mu(m) \in \alpha_\nu(x) \\
\equiv & \quad \{ \text{ calculus } \} \\
 & \mu(m) \in ([\![V]\!] \cap \alpha_\nu(x)) \\
\equiv & \quad \{ [\![V]\!] \cap \alpha_\nu(x) = \emptyset \} \\
 & \mathit{false}
\end{array}
$$

C.3 *VA3* $\nabla_V(\nabla_{V'}(x)) \doteq \nabla_{V \cap V'}(x)$

$$
\begin{array}{ll}
 & \mu(m) \in [\![V]\!] \wedge \mu(m) \in [\![V']\!] \\
\equiv & \quad \{ \text{ calculus } \}
\end{array}
$$

$\mu(m) \in ([\![V]\!] \cap [\![V']\!])$

$\equiv$ { def. $\cap$ }

$\mu(m) \in [\![V \cap V']\!]$

C.4 *VA4* $\nabla_V(x \parallel y) \doteq \nabla_V(x \parallel \nabla_{V'}(y))$ *if* $\Downarrow(V) \subseteq V'$

We define E as $\nabla_V(p) R \nabla_V(\nabla_{V'}(p))$ (for all $p \in T_{pc}$). The proof for this part is the same as the proof of axiom *VA3*, as $V \subseteq \Downarrow(V)$ and thus $V \cap V' = V$.

- $\nabla_V(p \parallel q) \xrightarrow{m} \surd$, which means that $p \parallel q \xrightarrow{m} \surd \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $p \xrightarrow{n} \surd \wedge q \xrightarrow{o} \surd \wedge m = n \oplus o$ and $o \in \Downarrow (V)$ and $\nabla_{V'}(q) \xrightarrow{o} \surd$ and therefore $p \parallel \nabla_{V'}(q) \xrightarrow{m} \surd$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \surd$
- $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \surd$, which means that $p \parallel \nabla_{V'}(q) \xrightarrow{m} \surd \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and $p \xrightarrow{n} \surd \wedge \nabla_{V'}(q) \xrightarrow{o} \surd \wedge m = n \oplus o$ and $q \xrightarrow{o} \surd \wedge \mu(o) \in [\![V']\!] \cup \{[]\}$ and therefore $p \parallel q \xrightarrow{m} \surd$ and $\nabla_V(p \parallel q) \xrightarrow{m} \surd$
- $\nabla_V(p \parallel q) \xrightarrow{m} p'$, which means that $p \parallel q \xrightarrow{m} p'' \wedge p' = \nabla_V(p'') \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and
  - $p \xrightarrow{m} \surd \wedge p'' = q$, which means that $p \parallel \nabla_{V'}(q) \xrightarrow{m} \nabla_{V'}(q)$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(\nabla_{V'}(q))$, with $p' R \nabla_V(\nabla_{V'}(q))$, or
  - $q \xrightarrow{m} \surd \wedge p'' = p$, which means that $\nabla_{V'}(q) \xrightarrow{m} \surd$ and $p \parallel \nabla_{V'}(q) \xrightarrow{m} p$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(p)$, with $p' R \nabla_V(p)$, or
  - $p \xrightarrow{m} p''' \wedge p'' = p''' \parallel q$, which means that $p \parallel \nabla_{V'}(q) \xrightarrow{m} p''' \parallel \nabla_{V'}(q)$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(p''' \parallel \nabla_{V'}(q))$, with $p' R \nabla_V(p''' \parallel \nabla_{V'}(q))$, or
  - $q \xrightarrow{m} q' \wedge p'' = p \parallel q'$, which means that $\nabla_{V'}(q) \xrightarrow{m} \nabla_{V'}(q')$ and $p \parallel \nabla_{V'}(q) \xrightarrow{m} p \parallel \nabla_{V'}(q')$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(p \parallel \nabla_{V'}(q'))$, with $p' R \nabla_V(p \parallel \nabla_{V'}(q'))$, or
  - $p \xrightarrow{n} p''' \wedge q \xrightarrow{o} q' \wedge m = n \oplus o \wedge p'' = p''' \parallel q'$, which means that $\nabla_{V'}(q) \xrightarrow{o} \nabla_{V'}(q')$ and $p \parallel \nabla_{V'}(q) \xrightarrow{m} p''' \parallel \nabla_{V'}(q')$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(p''' \parallel \nabla_{V'}(q'))$, with $p' R \nabla_V(p''' \parallel \nabla_{V'}(q'))$, or
  - $p \xrightarrow{n} \surd \wedge q \xrightarrow{o} q' \wedge m = n \oplus o \wedge p'' = q'$, which means that $\nabla_{V'}(q) \xrightarrow{o} \nabla_{V'}(q')$ and $p \parallel \nabla_{V'}(q) \xrightarrow{m} \nabla_{V'}(q')$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(\nabla_{V'}(q'))$, with $p' R \nabla_V(\nabla_{V'}(q'))$, or
  - $p \xrightarrow{n} p''' \wedge q \xrightarrow{o} \surd \wedge m = n \oplus o \wedge p'' = p'''$, which means that $\nabla_{V'}(q) \xrightarrow{o} \surd$ and $p \parallel \nabla_{V'}(q) \xrightarrow{m} p'''$ and $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} \nabla_V(p''')$, with $p' R \nabla_V(p''')$
- $\nabla_V(p \parallel \nabla_{V'}(q)) \xrightarrow{m} p'$, which means that $p \parallel \nabla_{V'}(q) \xrightarrow{m} p'' \wedge p' = \nabla_V(p'') \wedge \mu(m) \in [\![V]\!] \cup \{[]\}$ and
  - $p \xrightarrow{m} \surd \wedge p'' = \nabla_{V'}(q)$, which means that $p \parallel q \xrightarrow{m} q$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(q)$, with $p' R \nabla_V(q)$, or
  - $\nabla_{V'}(q) \xrightarrow{m} \surd \wedge p'' = p$, which means that $q \xrightarrow{m} \surd \wedge \mu(m) \in [\![V']\!] \cup \{[]\}$ and $p \parallel q \xrightarrow{m} p$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(p)$, with $p' R \nabla_V(p)$, or
  - $p \xrightarrow{m} p''' \wedge p'' = p''' \parallel \nabla_{V'}(q)$, which means that $p \parallel q \xrightarrow{m} p''' \parallel q$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(p''' \parallel q)$, with $p' R \nabla_V(p''' \parallel q)$, or
  - $\nabla_{V'}(q) \xrightarrow{m} q' \wedge p'' = p \parallel q'$, which means that $q \xrightarrow{m} q'' \wedge q' = \nabla_{V'}(q'') \wedge \mu(m) \in [\![V']\!] \cup \{[]\}$ and $p \parallel q \xrightarrow{m} p \parallel q''$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(p \parallel q'')$, with $p' R \nabla_V(p \parallel q'')$, or
  - $p \xrightarrow{n} p''' \wedge \nabla_{V'}(q) \xrightarrow{o} q' \wedge m = n \oplus o \wedge p'' = p''' \parallel q'$, which means that $q \xrightarrow{o} q'' \wedge q' = \nabla_{V'}(q'') \wedge \mu(o) \in [\![V']\!] \cup \{[]\}$ and $p \parallel q \xrightarrow{m} p''' \parallel q''$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(p''' \parallel q'')$, with $p' R \nabla_V(p''' \parallel q'')$, or

- $p \xrightarrow{n} \surd \wedge \nabla_{V'}(q) \xrightarrow{o} q' \wedge m = n \oplus o \wedge p'' = q'$, which means that $q \xrightarrow{o} q'' \wedge q' = \nabla_{V'}(q'') \wedge \mu(o) \in [\![V']\!] \cup \{[]\}$ and $p \parallel q \xrightarrow{m} q''$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(q'')$, with $p' R \nabla_V(q'')$, or
- $p \xrightarrow{n} p''' \wedge \nabla_{V'}(q) \xrightarrow{o} \surd \wedge m = n \oplus o \wedge p'' = p'''$, which means that $q \xrightarrow{o} \surd \wedge \mu(o) \in [\![V']\!] \cup \{[]\}$ and $p \parallel q \xrightarrow{m} p'''$ and $\nabla_V(p \parallel q) \xrightarrow{m} \nabla_V(p''')$, with $p' R \nabla_V(p''')$

C.5 $CA1 \quad \Gamma_C(x) \doteq x \quad if\ dom([\![C]\!]) \cap \Downarrow(\alpha_\nu(x)) = \emptyset$

$$
\begin{array}{ll}
 & \gamma(m, [\![C]\!]) = m \\
\equiv & \quad \{ \text{ def. } \gamma \ \} \\
 & \neg \exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a \rangle \vee c = \langle b, \tau \rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))) \\
\equiv & \quad \{ \ \mu(n) \in \Downarrow(\alpha_\nu(x)) \ \} \\
 & \neg \exists_{n,o}(m = n \oplus o \wedge false) \\
\equiv & \quad \{ \text{ calculus } \} \\
 & \neg false \\
\equiv & \quad \{ \text{ calculus } \} \\
 & true
\end{array}
$$

C.6 $CA2 \quad \Gamma_C(\Gamma_{C'}(x)) \doteq \Gamma_{C \cup C'}(x) \quad if\ \mathcal{N}(dom([\![C]\!])) \cap \mathcal{N}(dom([\![C']\!])) = \emptyset \wedge \mathcal{N}(dom([\![C]\!])) \cap rng([\![C']\!]) = \emptyset$

Induction on the length of m and by the cases of γ (with C'). Case $|m| = 0$:

$$
\begin{array}{ll}
 & \gamma(\gamma(m, [\![C']\!]), [\![C]\!]) = \gamma(m, [\![C \cup C']\!]) \\
\equiv & \quad \{ \ |m| = 0 \ \} \\
 & \gamma(\gamma([], [\![C']\!]), [\![C]\!]) = \gamma([], [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ def. } \gamma \ \} \\
 & [] = [] \\
\equiv & \quad \{ \text{ refl. } = \} \\
 & true
\end{array}
$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,a \rangle \in C'}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$
\begin{array}{ll}
 & \gamma(\gamma(m, [\![C']\!]), [\![C]\!]) = \gamma(m, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ take } n \text{ and } o \text{ with } m = n \oplus o \wedge \exists_{\langle b,a \rangle \in C'}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))) \ \} \\
 & \gamma(\gamma(n \oplus o, [\![C']\!]), [\![C]\!]) = \gamma(n \oplus o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ def. } \gamma, \text{ def. } n \text{ and } o \ \} \\
 & \gamma([a(\vec{d})] \oplus \gamma(o, [\![C']\!]), [\![C]\!]) = [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \ a \in rng(C), \mathcal{N}(dom([\![C]\!])) \cap rng([\![C']\!]) = \emptyset \ \} \\
 & [a(\vec{d})] \oplus \gamma(\gamma(o, [\![C']\!]), [\![C]\!]) = [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ induction hypothesis } \} \\
 & [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) = [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ refl. } = \} \\
 & true
\end{array}
$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,\tau\rangle \in C'}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, which is similar to the previous case.
Case $|m| > 0$ and $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C'}((c = \langle b,a\rangle \vee c = \langle b,\tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$\begin{array}{ll}
 & \gamma(\gamma(m, [\![C']\!]), [\![C]\!]) = \gamma(m, [\![C \cup C']\!]) \\
\equiv & \quad \{ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C'}((c = \langle b,a\rangle \vee c = \langle b,\tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{def. } \gamma \ \} \\
 & \gamma(m, [\![C]\!]) = \gamma(m, [\![C \cup C']\!])
\end{array}$$

Again induction on m (with C). Case $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$\begin{array}{ll}
 & \gamma(m, [\![C]\!]) = \gamma(m, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ take } n \text{ and } o \text{ with } m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))) \ \} \\
 & \gamma(n \oplus o, [\![C]\!]) = \gamma(n \oplus o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ def. } \gamma, \text{ def. } n \text{ and } o,\ c \in [\![C]\!] \Rightarrow c \in [\![C \cup C']\!] \ \} \\
 & [a(\vec{d})] \oplus \gamma(o, [\![C]\!]) = [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ induction hypothesis } \} \\
 & [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) = [a(\vec{d})] \oplus \gamma(o, [\![C \cup C']\!]) \\
\equiv & \quad \{ \text{ refl. } = \} \\
 & \mathit{true}
\end{array}$$

Case $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,\tau\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, which is similar to the previous case.

$$\begin{array}{ll}
 & \gamma(m, [\![C]\!]) = \gamma(m, [\![C \cup C']\!]) \\
\equiv & \quad \{ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b,a\rangle \vee c = \langle b,\tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \\
 & \quad\ \ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C'}((c = \langle b,a\rangle \vee c = \langle b,\tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{def. } \gamma \ \} \\
 & m = m \\
\equiv & \quad \{ \text{ refl. } = \} \\
 & \mathit{true}
\end{array}$$

C.7 $CA3\ \ \Gamma_C(x \parallel y) \doteq x \parallel \Gamma_C(y)\ \ if\ \ \Downarrow(dom([\![C]\!])) \cap \Downarrow(\alpha_\nu(x)) = \emptyset$

- $\Gamma_C(p \parallel q) \xrightarrow{m} \surd$, which means that $p \parallel q \xrightarrow{n} \surd \wedge m = \gamma(n, [\![C]\!])$ and $p \xrightarrow{o} \surd \wedge q \xrightarrow{o'} \surd \wedge n = o \oplus o'$ and therefore $\Gamma_C(q) \xrightarrow{\gamma(o', [\![C]\!])} \surd$ and $p \parallel \Gamma_C(q) \xrightarrow{o \oplus \gamma(o', [\![C]\!])} \surd$, with $m = \gamma(n, [\![C]\!]) = \gamma(o \oplus o', [\![C]\!]) = o \oplus \gamma(o', [\![C]\!])$ as $\Downarrow(\{o\}) \subseteq \Downarrow(\alpha_\nu(p))$ and thus $\Downarrow(dom([\![C]\!])) \cap \Downarrow(\{o\} = \emptyset$

- $p \parallel \Gamma_C(q) \xrightarrow{m} \surd$, which means that $p \xrightarrow{n} \surd \wedge \Gamma_C(q) \xrightarrow{o} \surd \wedge m = n \oplus o$ and $q \xrightarrow{o'} \surd \wedge o' = \gamma(o, [\![C]\!])$ and therefore $p \parallel q \xrightarrow{n \oplus o'} \surd$ and $\Gamma_C(p \parallel q) \xrightarrow{\gamma(n \oplus o', [\![C]\!])} \surd$, with $m = n \oplus o = n \oplus \gamma(o', [\![C]\!]) = \gamma(n \oplus o', [\![C]\!])$ as $\Downarrow(\{n\}) \subseteq \Downarrow(\alpha_\nu(p))$

- $\Gamma_C(p \parallel q) \xrightarrow{m} p'$, which means that $p \parallel q \xrightarrow{n} p'' \wedge m = \gamma(n, [\![C]\!]) \wedge p' = \Gamma_C(p'')$ and
  - $p \xrightarrow{n} \surd \wedge p'' = q$ and therefore $p \parallel \Gamma_C(q) \xrightarrow{n} \Gamma_C(q)$, with $p' R \Gamma_C(q)$ and $m = \gamma(n, [\![C]\!]) = n$ as $\Downarrow(\{n\}) \subseteq \Downarrow(\alpha_\nu(p))$, or
  - $q \xrightarrow{n} \surd \wedge p'' = p$ and therefore $\Gamma_C(q) \xrightarrow{\gamma(n, [\![C]\!])} \surd$ and $p \parallel \Gamma_C(q) \xrightarrow{\gamma(n, [\![C]\!])} p$, with $p' R p$, or
  - $p \xrightarrow{n} p''' \wedge p'' = p''' \parallel q$ and therefore $p \parallel \Gamma_C(q) \xrightarrow{n} p''' \parallel \Gamma_C(q)$, with $p' R p''' \parallel \Gamma_C(q)$ and $m = \gamma(n, [\![C]\!]) = n$ as $\Downarrow(\{n\}) \subseteq \Downarrow(\alpha_\nu(p))$, or

  - $q \xrightarrow{n} q' \wedge p'' = p \parallel q'$ and therefore $\Gamma_C(q) \xrightarrow{\gamma(n,[\![C]\!])} \Gamma_C(q')$ and $p \parallel \Gamma_C(q) \xrightarrow{\gamma(n,[\![C]\!])} p \parallel \Gamma_C(q')$, with $p' R p \parallel \Gamma_C(q')$, or
  - $p \xrightarrow{o} p''' \wedge q \xrightarrow{o'} q' \wedge n = o \oplus o' \wedge p'' = p''' \parallel q'$ and therefore $\Gamma_C(q) \xrightarrow{\gamma(o',[\![C]\!])} \Gamma_C(q')$ and $p \parallel \Gamma_C(q) \xrightarrow{o \oplus \gamma(o',[\![C]\!])} p''' \parallel \Gamma_C(q')$, with $p' R p''' \parallel \Gamma_C(q')$ and $m = \gamma(n, repC) = \gamma(o \oplus o', [\![C]\!]) = o \oplus \gamma(o', [\![C]\!])$ as $\Downarrow(\{o\}) \subseteq \Downarrow(\alpha_\nu(p))$, or
  - $p \xrightarrow{o} \checkmark \wedge q \xrightarrow{o'} q' \wedge n = o \oplus o' \wedge p'' = q'$ and therefore $\Gamma_C(q) \xrightarrow{\gamma(o',[\![C]\!])} \Gamma_C(q')$ and $p \parallel \Gamma_C(q) \xrightarrow{o \oplus \gamma(o',[\![C]\!])} \Gamma_C(q')$, with $p' R \Gamma_C(q')$ and $m = \gamma(n, repC) = \gamma(o \oplus o', [\![C]\!]) = o \oplus \gamma(o', [\![C]\!])$ as $\Downarrow(\{o\}) \subseteq \Downarrow(\alpha_\nu(p))$, or
  - $p \xrightarrow{o} p''' \wedge q \xrightarrow{o'} \checkmark \wedge n = o \oplus o' \wedge p'' = p'''$ and therefore $\Gamma_C(q) \xrightarrow{\gamma(o',[\![C]\!])} \checkmark$ and $p \parallel \Gamma_C(q) \xrightarrow{o \oplus \gamma(o',[\![C]\!])} p'''$, with $p' R p'''$ and $m = \gamma(n, [\![C]\!]) = \gamma(o \oplus o', [\![C]\!]) = o \oplus \gamma(o', [\![C]\!])$ as $\Downarrow(\{o\}) \subseteq \Downarrow(\alpha_\nu(p))$

- $p \parallel \Gamma_C(q) \xrightarrow{m} p'$, which is similar to the previous case

C.8 *CA4* $\Gamma_C(x \parallel y) \doteq \Gamma_C(x \parallel \Gamma_C(y))$ *if* $\mathcal{N}(dom([\![C]\!])) \cap rng([\![C]\!]) = \emptyset$

With (implicit use of) Lemma C.2, the soundness proof of the axiom becomes similar to the proof of axiom *VA*4.

C.9 *DA*1 $\partial_H(x) \doteq x$ *if* $[\![H]\!] \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset$

$$
\begin{array}{ll}
 & \mu(m) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ Lemma C.1 } \} \\
 & \mathcal{N}(\{\mu(m)\}) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \{\mu(m)\} \subseteq \alpha_\nu(x),\ [\![H]\!] \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset \} \\
 & true
\end{array}
$$

C.10 *DA*2 $\partial_H(x) \doteq \delta$ *if* $\forall_{v \in \alpha_\nu(x)}(v \cap [\![H]\!] \neq \emptyset)$

$$
\begin{array}{ll}
 & \mu(m) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \alpha_\nu,\ \forall_{v \in \alpha_\nu(x)}(v \cap [\![H]\!] \neq \emptyset) \} \\
 & false
\end{array}
$$

C.11 *DA*3 $\partial_H(\partial_{H'}(x)) \doteq \partial_{H \cup H'}(x)$

$$
\begin{array}{ll}
 & \mu(m) \cap [\![H]\!] = \emptyset \wedge \mu(m) \cap [\![H']\!] = \emptyset \\
\equiv & \quad \{ \text{ set calculus } \} \\
 & (\mu(m) \cap [\![H]\!]) \cup (\mu(m) \cap [\![H']\!]) = \emptyset \\
\equiv & \quad \{ \text{ set calculus } \} \\
 & \mu(m) \cap ([\![H]\!] \cup [\![H']\!]) = \emptyset \\
\equiv & \quad \{ \text{ def. } \cup \} \\
 & \mu(m) \cap ([\![H \cup H']\!]) = \emptyset
\end{array}
$$

C.12 *DA4* $\partial_H(x \parallel y) \doteq \partial_H(x) \parallel \partial_H(y)$

- $\partial_H(p \parallel q)\xrightarrow{m}\surd$, which means that $p \parallel q\xrightarrow{m}\surd \wedge \mu(m) \cap [\![H]\!] = \emptyset$ and $p\xrightarrow{n}\surd \wedge q\xrightarrow{o}\surd \wedge m = n \oplus o$ and therefore $\partial_H(p)\xrightarrow{n}\surd \wedge \partial_H(q)\xrightarrow{o}\surd$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\surd$

- $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\surd$, which means that $\partial_H(p)\xrightarrow{n}\surd \wedge \partial_H(q)\xrightarrow{o}\surd \wedge m = n \oplus o$ and $p\xrightarrow{n}\surd \wedge q\xrightarrow{o}\surd \wedge \mu(n) \cap [\![H]\!] = \emptyset \wedge \mu(o) \cap [\![H]\!] = \emptyset$ and therefore $p \parallel q\xrightarrow{m}\surd$ and $\partial_H(p \parallel q)\xrightarrow{m}\surd$

- $\partial_H(p \parallel q)\xrightarrow{m}p'$, which means that $p \parallel q\xrightarrow{m}p'' \wedge p' = \partial_H(p'') \wedge \mu(m) \cap [\![H]\!] = \emptyset$ and

  - $p\xrightarrow{m}\surd \wedge p'' = q$ and therefore $\partial_H(p)\xrightarrow{m}\surd$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(q)$, with $p'R\partial_H(q)$, or
  - $q\xrightarrow{m}\surd \wedge p'' = p$ and therefore $\partial_H(q)\xrightarrow{m}\surd$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(p)$, with $p'R\partial_H(p)$, or
  - $p\xrightarrow{m}p''' \wedge p'' = p''' \parallel q$ and therefore $\partial_H(p)\xrightarrow{m}\partial_H(p''')$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(p''') \parallel \partial_H(q)$, with $p'R\partial_H(p''') \parallel \partial_H(q)$, or
  - $q\xrightarrow{m}q' \wedge p'' = p \parallel q'$ and therefore $\partial_H(q)\xrightarrow{m}\partial_H(q')$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(p) \parallel \partial_H(q')$, with $p'R\partial_H(p) \parallel \partial_H(q')$, or
  - $p\xrightarrow{n}p''' \wedge q\xrightarrow{o}q' \wedge m = n \oplus o \wedge p'' = p''' \parallel q'$ and therefore $\partial_H(p)\xrightarrow{n}\partial_H(p''') \wedge \partial_H(q)\xrightarrow{o}\partial_H(q')$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(p''') \parallel \partial_H(q')$, with $p'R\partial_H(p''') \parallel \partial_H(q')$, or
  - $p\xrightarrow{n}\surd \wedge q\xrightarrow{o}q' \wedge m = n \oplus o \wedge p'' = q'$ and therefore $\partial_H(p)\xrightarrow{n}\surd \wedge \partial_H(q)\xrightarrow{o}\partial_H(q')$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(q')$, with $p'R\partial_H(q')$, or
  - $p\xrightarrow{n}p''' \wedge q\xrightarrow{o}\surd \wedge m = n \oplus o \wedge p'' = p'''$ and therefore $\partial_H(p)\xrightarrow{n}\partial_H(p''') \wedge \partial_H(q)\xrightarrow{o}\surd$ and $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}\partial_H(p''')$, with $p'R\partial_H(p''')$

- $\partial_H(p) \parallel \partial_H(q)\xrightarrow{m}p'$, which is similar to the previous case

C.13 *TA1* $\tau_I(x) \doteq x$ *if* $[\![I]\!] \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset$

We prove $\theta(m, [\![I]\!]) = m$ with induction on the structure of m. Case $m = []$:

$$
\begin{array}{ll}
 & \theta([], [\![I]\!]) \\
= & \quad \{ \text{ def. } \theta \ \} \\
 & []
\end{array}
$$

Case $m = [a(\vec{d})] \oplus n$:

$$
\begin{array}{ll}
 & \theta([a(\vec{d})] \oplus n, [\![I]\!]) \\
= & \quad \{ \text{ def. } \theta,\ a \notin [\![I]\!] \ \} \\
 & [a(\vec{d})] \oplus \theta(n, [\![I]\!]) \\
= & \quad \{ \text{ induction hypothesis } \} \\
 & [a(\vec{d})] \oplus n
\end{array}
$$

C.14 *TA3* $\tau_I(\tau_{I'}(x)) \doteq \tau_{I\cup I'}(x)$

We prove $\theta(\theta(m, [\![I']\!]), [\![I]\!]) = \theta(m, [\![I \cup I']\!])$ with induction on the structure of m. Case $m = []$:

$$\begin{array}{ll}
 & \theta(\theta([], [\![I']\!]), [\![I]\!]) \\
= & \quad \{ \text{ def. } \theta \} \\
 & [] \\
= & \quad \{ \text{ def. } \theta \} \\
 & \theta([], [\![I \cup I']\!])
\end{array}$$

Case $m = [a(\vec{d})] \oplus n$:

$$\begin{array}{lll}
 & \theta(\theta([a(\vec{d})] \oplus n, [\![I']\!]), [\![I]\!]) & \\
= & \quad \{ \text{ def. } \theta \} & \\
 & \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \in [\![I']\!] \\
 & \theta([a(\vec{d})] \oplus \theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \notin [\![I']\!] \\
= & \quad \{ \text{ def. } \theta \} & \\
 & \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \in [\![I']\!] \\
 & \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \notin [\![I']\!] \wedge a \in [\![I]\!] \\
 & [a(\vec{d})] \oplus \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \notin [\![I']\!] \wedge a \notin [\![I]\!] \\
= & \quad \{ \text{ calculus } \} & \\
 & \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \in [\![I']\!] \vee a \in [\![I]\!] \\
 & [a(\vec{d})] \oplus \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \notin [\![I']\!] \wedge a \notin [\![I]\!] \\
= & \quad \{ \text{ calculus } \} & \\
 & \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \in [\![I \cup I']\!] \\
 & [a(\vec{d})] \oplus \theta(\theta(n, [\![I']\!]), [\![I]\!]) & \textit{if } a \notin [\![I \cup I']\!] \\
= & \quad \{ \text{ induction hypothesis } \} & \\
 & \theta(n, [\![I \cup I']\!]) & \textit{if } a \in [\![I \cup I']\!] \\
 & [a(\vec{d})] \oplus \theta(n, [\![I \cup I']\!]) & \textit{if } a \notin [\![I \cup I']\!] \\
= & \quad \{ \text{ def. } \theta \} & \\
 & \theta([a(\vec{d})] \oplus n, [\![I \cup I']\!]) &
\end{array}$$

C.15 *TA4* $\tau_I(x \parallel y) \doteq \tau_I(x) \parallel \tau_I(y)$

This proof is similar to the proof of axiom *DA*4.

C.16 *RA*1 $\rho_R(x) \doteq x \quad \textit{if } \textit{dom}([\![R]\!]) \cap \mathcal{N}(\alpha_\nu(x)) = \emptyset$

We prove $[\![R]\!] \bullet m = m$ with induction on the structure of m. Case $m = []$:

$$\begin{array}{ll}
 & [\![R]\!] \bullet [] \\
= & \quad \{ \text{ def. } \bullet \} \\
 & []
\end{array}$$

Case $m = [a(\vec{d})] \oplus n$:

$$\begin{array}{ll}
 & [\![R]\!] \bullet ([a(\vec{d})] \oplus n) \\
= & \quad \{ \text{ def. } \bullet \}
\end{array}$$

$$
\begin{array}{ll}
 & [[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet n) \\
= & \quad \{ \text{ induction hypothesis } \} \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus n \\
= & \quad \{ a \notin dom([\![R]\!]) \} \\
 & [a(\vec{d})] \oplus n
\end{array}
$$

C.17 *RA2* $\rho_R(\rho_{R'}(x)) \doteq \rho_{R\cup R'}(x)$ *if* $dom([\![R]\!]) \cap dom([\![R']\!]) = \emptyset \wedge dom([\![R]\!]) \cap rng([\![R']\!]) = \emptyset$

This is a special case of axiom *RA*3, as we have the following:

$$
\begin{array}{ll}
 & \{\langle a,b\rangle \mid ((\langle a,b\rangle \in [\![R]\!] \wedge a \notin (dom([\![R']\!]) \cup rng([\![R']\!]))) \vee ((\langle c,b\rangle \in [\![R]\!] \wedge \langle a,c\rangle \in [\![R']\!]) \vee \\
 & \qquad ((\langle a,b\rangle \in [\![R']\!] \wedge b \notin dom([\![R]\!]))\} \\
\equiv & \quad \{ dom([\![R]\!]) \cap dom([\![R']\!]) = \emptyset \wedge dom([\![R]\!]) \cap rng([\![R']\!]) = \emptyset \} \\
 & \{\langle a,b\rangle \mid ((\langle a,b\rangle \in [\![R]\!] \wedge true) \vee false \vee ((\langle a,b\rangle \in [\![R']\!] \wedge true)\} \\
\equiv & \quad \{ \text{ calculus } \} \\
 & \{\langle a,b\rangle \mid \langle a,b\rangle \in [\![R]\!] \vee \langle a,b\rangle \in [\![R']\!]\} \\
\equiv & \quad \{ \text{ calculus } \} \\
 & [\![R \cup R']\!]
\end{array}
$$

C.18 *RA3* $\rho_R(\rho_{R'}(x)) \doteq \rho_{R''}(x)$ *if* $[\![R'']\!] = \{\langle a,b\rangle \mid ((\langle a,b\rangle \in [\![R]\!] \wedge a \notin (dom([\![R']\!]) \cup rng([\![R']\!]))) \vee ((\langle c,b\rangle \in [\![R]\!] \wedge \langle a,c\rangle \in [\![R']\!]) \vee ((\langle a,b\rangle \in [\![R']\!] \wedge b \notin dom([\![R]\!]))\}$

We prove $[\![R]\!] \bullet ([\![R']\!] \bullet m) = [\![R'']\!] \bullet m$ with induction on the structure of m. Case $m = []$:

$$
\begin{array}{ll}
 & [\![R]\!] \bullet ([\![R']\!] \bullet []) \\
= & \quad \{ \text{ def. } \bullet \} \\
 & [] \\
= & \quad \{ \text{ def. } \bullet \} \\
 & [\![R'']\!] \bullet []
\end{array}
$$

Case $m = [a(\vec{d})] \oplus n$:

$$
\begin{array}{lll}
 & [\![R]\!] \bullet ([\![R']\!] \bullet ([a(\vec{d})] \oplus n)) & \\
= & \quad \{ \text{ def. } \bullet \} & \\
 & [[\![R]\!]^+([\![R']\!]^+(a))(\vec{d})] \oplus ([\![R]\!] \bullet ([\![R']\!] \bullet n)) & \\
= & \quad \{ \text{ induction hypothesis } \} & \\
 & [[\![R]\!]^+([\![R']\!]^+(a))(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \\
= & \quad \{ \text{ case analysis } [\![R']\!]^+(a) \} & \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \text{if } a \notin dom(R') \\
 & [[\![R]\!]^+([\![R']\!]^+(a))(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \text{if } a \in dom(R') \\
= & \quad \{ \text{ case analysis } [\![R]\!]^+([\![R']\!]^+(a)) \} & \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \text{if } a \notin dom(R') \\
 & [[\![R']\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \text{if } a \in dom(R') \wedge [\![R']\!]^+(a) \notin dom(R) \\
 & [[\![R]\!]^+([\![R']\!]^+(a))(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \text{if } a \in dom(R') \wedge [\![R']\!]^+(a) \in dom(R) \\
= & \quad \{ \text{ def. } R'' \} &
\end{array}
$$

$$\begin{array}{lll}
 & [[\![R'']\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \textit{if } a \notin dom(R') \\
 & [[\![R'']\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \textit{if } a \in dom(R') \wedge [\![R']\!]^+(a) \notin dom(R) \\
 & [[\![R'']\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \textit{if } a \in dom(R') \wedge [\![R']\!]^+(a) \in dom(R) \\
= & \quad \{ \text{ case elimination } \} & \\
 & [[\![R'']\!]^+(a)(\vec{d})] \oplus ([\![R'']\!] \bullet n) & \\
= & \quad \{ \text{ def. } \bullet \text{ } \} & \\
 & [\![R'']\!] \bullet ([a(\vec{d})] \oplus n) &
\end{array}$$

C.19 $RA4 \quad \rho_R(x \parallel y) \doteq \rho_R(x) \parallel \rho_R(y)$

This proof is similar to the proof of axiom $DA4$.

C.20 $VC1 \quad \nabla_V(\Gamma_C(x)) \doteq \nabla_V(\gamma_C(\nabla_{V'}(x))) \quad \textit{if } [\![V']\!] = \{v \oplus w \mid \gamma(v, [\![C]\!]) \oplus w \in [\![V]\!]\}$

We prove $\mu(\gamma(m, [\![C]\!])) \in [\![V]\!] \cup \{[]\} \Rightarrow \mu([]) \in [\![V']\!] \cup \{[]\}$ (as $(P \equiv Q \wedge P) \equiv (P \Rightarrow Q)$) by induction on the length of m and by the cases of γ. Case $|m| = 0$:

$$\begin{array}{ll}
 & \mu(\gamma([], [\![C]\!])) \in [\![V]\!] \cup \{[]\} \\
\Rightarrow & \quad \{ \text{ right zero } \Rightarrow \text{ } \} \\
 & \textit{true} \\
\Rightarrow & \quad \{ \mu([]) = [] \} \\
 & \mu([]) \in [\![V']\!] \cup \{[]\}
\end{array}$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$\begin{array}{ll}
 & \mu(\gamma(m, [\![C]\!])) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{ \text{ take } n \text{ and } o \text{ with } m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))) \text{ } \} \\
 & \mu(\gamma(n \oplus o, [\![C]\!])) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{ \text{ take } \langle b, a\rangle \in C \text{ with } b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{ def. } \gamma \text{ } \} \\
 & \mu([a(\vec{d})] \oplus \gamma(o, [\![C]\!])) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{ \text{ def. } \mu \text{ } \} \\
 & [a] \oplus \mu(\gamma(o, [\![C]\!])) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{ \text{ calculus } \} \\
 & [a] \oplus \mu(\gamma(o, [\![C]\!])) \in \{v \mid v \in [\![V]\!]\} \\
\equiv & \quad \{ \text{ calculus } \} \\
 & [a] \oplus \mu(\gamma(o, [\![C]\!])) \in \{[a] \oplus v \mid [a] \oplus v \in [\![V]\!]\} \\
\equiv & \quad \{ \text{ calculus } \} \\
 & \mu(\gamma(o, [\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \\
\equiv & \quad \{ \text{ idem. } \wedge \text{ } \} \\
 & \mu(\gamma(o, [\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(\gamma(o, [\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \\
\Rightarrow & \quad \{ \text{ induction hypothesis } \} \\
 & \mu(\gamma(o, [\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{v \oplus w \mid \gamma(v, [\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\} \cup \{[]\} \\
\equiv & \quad \{ \text{ calculus } \}
\end{array}$$

$$
\begin{array}{ll}
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\})\vee \\
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{[]\}) \\
\equiv & \{ \text{ calculus, } \mu(o) \in \{[]\} \equiv o = [] \ \} \\
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\})\vee \\
 & (\mu(\gamma([],[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge o = []) \\
\equiv & \{ \text{ calculus, } \mu(\gamma([],[\![C]\!])) = [] \ \} \\
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\})\vee \\
 & ([a] \in [\![V]\!] \wedge o = []) \\
\equiv & \{ \text{ calculus } \} \\
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\})\vee \\
 & (\mu(o) \in \{v' \mid [a] \oplus v' \in [\![V]\!]\} \wedge o = []) \\
\equiv & \{ \text{ calculus } \} \\
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \wedge \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\})\vee \\
 & (\mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\} \wedge o = []) \\
\equiv & \{ \text{ calculus } \} \\
 & (\mu(\gamma(o,[\![C]\!])) \in \{v \mid [a] \oplus v \in [\![V]\!]\} \vee (\mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\} \wedge o = []))\wedge \\
 & (\mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\}\vee \\
 & \ (\mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\} \wedge o = [])) \\
\Rightarrow & \{ \text{ weakening } \} \\
 & \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\}\vee \\
 & (\mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\} \wedge o = []) \\
\equiv & \{ P \vee (P \wedge Q) \equiv P \ \} \\
 & \mu(o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in \{v' \mid [a] \oplus v' \in [\![V]\!]\}\} \\
\equiv & \{ \text{ calculus } \} \\
 & \mu(o) \in \{v \oplus w \mid [a] \oplus \gamma(v,[\![C]\!]) \oplus w \in \{[a] \oplus v' \mid [a] \oplus v' \in [\![V]\!]\}\} \\
\Rightarrow & \{ \text{ calculus, def. } \gamma \text{ and } a \ \} \\
 & \mu(o) \in \{v \oplus w \mid \gamma(\mu(n) \oplus v,[\![C]\!]) \oplus w \in \{v' \mid v' \in [\![V]\!]\}\} \\
\equiv & \{ \text{ calculus } \} \\
 & \mu(n) \oplus \mu(o) \in \{\mu(n) \oplus v \oplus w \mid \gamma(\mu(n) \oplus v,[\![C]\!]) \oplus w \in [\![V]\!]\} \\
\Rightarrow & \{ \text{ calculus, def. } \mu \ \} \\
 & \mu(n \oplus o) \in \{v \oplus w \mid \gamma(v,[\![C]\!]) \oplus w \in [\![V]\!]\} \\
\equiv & \{ \text{ def. } n, o \text{ and } V' \ \} \\
 & \mu(m) \in [\![V']\!] \\
\Rightarrow & \{ \text{ weakening } \} \\
 & \mu(m) \in [\![V']\!] \cup \{[]\}
\end{array}
$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,\tau\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, which is similar to the previous case.
Case $|m| > 0$ and $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b,a\rangle \vee c = \langle b,\tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$
\begin{array}{ll}
 & \mu(\gamma(m,[\![C]\!])) \in [\![V]\!] \cup \{[]\} \\
\equiv & \{ \ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b,a\rangle \vee c = \langle b,\tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{ def. } \gamma \ \}
\end{array}
$$

$$\begin{array}{ll}
 & \mu(m) \in [\![V]\!] \cup \{[]\} \\
\Rightarrow & \quad \{\ [\![V]\!] \subseteq [\![V']\!]\ \} \\
 & \mu(m) \in [\![V']\!] \cup \{[]\}
\end{array}$$

C.21 $VC2\quad \Gamma_C(\nabla_V(x)) \doteq \nabla_V(x) \quad \textit{if } dom(C) \cap \Downarrow(V) = \emptyset$

$$\begin{array}{ll}
 & \mu(m) \in [\![V]\!] \\
\Rightarrow & \quad \{\ \text{def. } \Downarrow\ \} \\
 & \Downarrow(\{\mu(m)\}) \subseteq \Downarrow(V) \\
\Rightarrow & \quad \{\ dom(C) \cap \Downarrow(V) = \emptyset\ \} \\
 & dom(C) \cap \Downarrow(\{\mu(m)\}) = \emptyset \\
\Rightarrow & \quad \{\ \text{calculus}\ \} \\
 & \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))) \\
\Rightarrow & \quad \{\ \text{def. } \gamma\ \} \\
 & \gamma(m, [\![C]\!]) = m
\end{array}$$

C.22 $VD1\quad \nabla_V(\partial_H(x)) \doteq \partial_H(\nabla_V(x))$

$$\begin{array}{ll}
 & \mu(m) \cap [\![H]\!] = \emptyset \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{\ \text{comm. } \wedge\ \} \\
 & \mu(m) \in [\![V]\!] \cup \{[]\} \wedge \mu(m) \cap [\![H]\!] = \emptyset
\end{array}$$

C.23 $VD2\quad \nabla_V(\partial_H(x)) \doteq \nabla_{V'}(x) \quad \textit{if } [\![V']\!] = \{v \mid v \in [\![V]\!] \wedge \mathcal{N}(\{v\}) \cap [\![H]\!] = \emptyset\}$

$$\begin{array}{ll}
 & \mu(m) \cap [\![H]\!] = \emptyset \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{\ \text{Lemma C.1}\ \} \\
 & \mathcal{N}(\{\mu(m)\}) \cap [\![H]\!] = \emptyset \wedge \mu(m) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{\ \text{calculus}\ \} \\
 & \mu(m) \in \{v \mid v \in [\![V]\!] \wedge \mathcal{N}(\{v\}) \cap [\![H]\!] = \emptyset\} \cup \{[]\} \\
\equiv & \quad \{\ \text{def. } V'\ \} \\
 & \mu(m) \in [\![V']\!] \cup \{[]\}
\end{array}$$

C.24 $VD3\quad \partial_H(\nabla_V(x)) \doteq \nabla_{V'}(x) \quad \textit{if } [\![V']\!] = \{v \mid v \in [\![V]\!] \wedge \mathcal{N}(\{v\}) \cap [\![H]\!] = \emptyset\}$

This proof is similar to the proof of axiom *VD*2.

C.25 $VT\quad \nabla_V(\tau_I(x)) \doteq \tau_I(\nabla_{V'}(x)) \quad \textit{if } [\![V']\!] = \{v \mid \theta(v, [\![I]\!]) \in [\![V]\!]\}$

$$\begin{array}{ll}
 & \mu(\theta(m, [\![I]\!])) \in [\![V]\!] \cup \{[]\} \\
\equiv & \quad \{\ \text{calculus}\ \} \\
 & \mu(\theta(m, [\![I]\!])) \in \{v \mid v \in [\![V]\!]\} \cup \{[]\} \\
\equiv & \quad \{\ \text{calculus}\ \} \\
 & \mu(\theta(m, [\![I]\!])) \in \{\theta(v, [\![I]\!]) \mid \theta(v, [\![I]\!]) \in [\![V]\!]\} \cup \{[]\} \\
\equiv & \quad \{\ \text{calculus}\ \}
\end{array}$$

$\mu(m) \in \{v \mid \theta(v, [\![I]\!]) \in [\![V]\!]\} \cup \{[]\}$
$\equiv$ { def. V' }
$\mu(m) \in [\![V']\!] \cup \{[]\}$

C.26 *VR* $\nabla_V(\rho_R(x)) \doteq \rho_R(\nabla_{V'}(x))$ *if* $[\![V']\!] = \{v \mid ([\![R]\!] \bullet v) \in [\![V]\!]\}$

$\mu([\![R]\!] \bullet m) \in [\![V]\!] \cup \{[]\}$
$\equiv$ { calculus }
$\mu([\![R]\!] \bullet m) \in \{v \mid v \in [\![V]\!]\} \cup \{[]\}$
$\equiv$ { calculus }
$\mu([\![R]\!] \bullet m) \in \{[\![R]\!] \bullet v \mid ([\![R]\!] \bullet v) \in [\![V]\!]\} \cup \{[]\}$
$\equiv$ { calculus }
$\mu(m) \in \{v \mid ([\![R]\!] \bullet v) \in [\![V]\!]\} \cup \{[]\}$
$\equiv$ { def. V' }
$\mu(m) \in [\![V']\!] \cup \{[]\}$

C.27 *CD*1 $\partial_H(\Gamma_C(x)) \doteq \Gamma_C(\partial_H(x))$ *if* $(\mathcal{N}(dom(C)) \cup rng(C)) \cap H = \emptyset$

We prove $\mu(\gamma(m, [\![C]\!])) \cap H = \emptyset \equiv \mu(m) \cap H = \emptyset$ by induction on the length of m and by the cases of γ. Case $|m| = 0$:

$\mu(\gamma([], [\![C]\!])) \cap H = \emptyset$
$\equiv$ { def. γ }
$\mu([]) \cap H = \emptyset$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$\mu(\gamma(m, [\![C]\!])) \cap H = \emptyset$
$\equiv$ { take n and o with $m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$ }
$\mu(\gamma(n \oplus o, [\![C]\!])) \cap H = \emptyset$
$\equiv$ { take $\langle b, a\rangle \in C$ with $b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))$), def. γ }
$\mu([a(\vec{d})] \oplus \gamma(o, [\![C]\!])) \cap H = \emptyset$
$\equiv$ { Lemma C.1 }
$\mathcal{N}(\{\mu([a(\vec{d})] \oplus \gamma(o, [\![C]\!]))\}) \cap H = \emptyset$
$\equiv$ { def. $\mathcal{N}$ }
$(\{a\} \cup \mathcal{N}(\{\mu(\gamma(o, [\![C]\!]))\})) \cap H = \emptyset$
$\equiv$ { Lemma C.1, calculus }
$\{a\} \cap H = \emptyset \wedge \mu(\gamma(o, [\![C]\!])) \cap H = \emptyset$
$\equiv$ { induction hypothesis }
$\{a\} \cap H = \emptyset \wedge \mu(o) \cap H = \emptyset$
$\equiv$ { $a \in rng(C)$ }
$\mu(o) \cap H = \emptyset$
$\equiv$ { $\mu(n) \in dom(C)$ }

$\mu(n) \cap H = \emptyset \wedge \mu(o) \cap H = \emptyset$
$\equiv$ { Lemma C.1, calculus }
$\mu(n \oplus o) \cap H = \emptyset$
$\equiv$ { def. n and o }
$\mu(m) \cap H = \emptyset$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,\tau\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, which is similar to the previous case.
Case $|m| > 0$ and $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$\mu(\gamma(m, [\![C]\!])) \cap H = \emptyset$
$\equiv$ { $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, def. γ }
$\mu(m) \cap H = \emptyset$

C.28 *CD*2 $\Gamma_C(\partial_H(x)) \doteq \partial_H(x)$ *if* $\mathcal{N}(dom(C)) \subseteq H$

$\mu(m) \cap H = \emptyset$
$\equiv$ { Lemma C.1 }
$\mathcal{N}(\{\mu(m)\}) \cap H = \emptyset$
$\Rightarrow$ { $\mathcal{N}(dom(C)) \subseteq H$ }
$\mathcal{N}(\{\mu(m)\}) \cap \mathcal{N}(dom(C)) = \emptyset$
$\Rightarrow$ { $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, def. γ }
$\gamma(m, [\![C]\!]) = m$

C.29 *CT*1 $\tau_I(\Gamma_C(x)) \doteq \Gamma_C(\tau_I(x))$ *if* $(\mathcal{N}(dom(C)) \cup rng(C)) \cap I = \emptyset$

We prove $\theta(\gamma(m, [\![C]\!]), [\![I]\!]) = \gamma(\theta(m, [\![I]\!]), [\![C]\!])$ by induction on the length of m and by the cases of γ.
Case $|m| = 0$:

$\theta(\gamma([], [\![C]\!]), [\![I]\!])$
$=$ { def. γ and θ }
$[]$
$=$ { def. γ and θ }
$\gamma(\theta([], [\![I]\!]), [\![C]\!])$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$\theta(\gamma(m, [\![C]\!]), [\![I]\!])$
$=$ { take n and o with $m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$ }
$\theta(\gamma(n \oplus o, [\![C]\!]), [\![I]\!])$
$=$ { take $\langle b, a\rangle \in C$ with $b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, def. γ }
$\theta([a(\vec{d})] \oplus \gamma(o, [\![C]\!]), [\![I]\!])$
$=$ { $a \in rng(C)$ }
$[a(\vec{d})] \oplus \theta(\gamma(o, [\![C]\!]), [\![I]\!])$
$=$ { induction hypothesis }

$$\begin{array}{ll}
 & [a(\vec{d})] \oplus \gamma(\theta(o, [\![I]\!]), [\![C]\!]) \\
= & \quad \{ \text{ def. } a \text{ and } \gamma \} \\
 & \gamma(n \oplus \theta(o, [\![I]\!]), [\![C]\!]) \\
= & \quad \{ n \in dom(C) \} \\
 & \gamma(\theta(n \oplus o, [\![I]\!]), [\![C]\!]) \\
= & \quad \{ \text{ def. } n \text{ and } o \} \\
 & \gamma(\theta(m, [\![I]\!]), [\![C]\!])
\end{array}$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,\tau\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, which is similar to the previous case.
Case $|m| > 0$ and $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$\begin{array}{ll}
 & \theta(\gamma(m, [\![C]\!]), [\![I]\!]) \\
= & \quad \{ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{ def. } \gamma \} \\
 & \theta(m, [\![I]\!]) \\
= & \quad \{ \theta(m, [\![I]\!]) \subseteq m, \text{ def. } \gamma \} \\
 & \gamma(\theta(m, [\![I]\!]), [\![C]\!])
\end{array}$$

C.30 *CT2* $\Gamma_C(\tau_I(x)) \doteq \tau_I(x)$ *if* $\mathcal{N}(dom(C)) \subseteq I$

$$\begin{array}{ll}
 & \gamma(\theta(m, [\![I]\!]), [\![C]\!]) = \theta(m, [\![I]\!]) \\
\equiv & \quad \{ \text{ def. } \gamma \} \\
 & \neg\exists_{n,o}(\theta(m, [\![I]\!]) = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))) \\
\equiv & \quad \{ \theta(m, [\![I]\!]) \cap [\![I]\!] = \emptyset \} \\
 & \mathit{true}
\end{array}$$

C.31 *CR1* $\rho_R(\Gamma_C(x)) \doteq \Gamma_C(\rho_R(x))$ *if* $dom(R) \cap rng(C) = \emptyset \wedge dom(R) \cap \mathcal{N}(dom(C)) = \emptyset \wedge rng(R) \cap \mathcal{N}(dom(C)) = \emptyset$

We prove $[\![R]\!] \bullet \gamma(m, [\![C]\!]) = \gamma([\![R]\!] \bullet m, [\![C]\!])$ by induction on the length of m and by the cases of γ.
Case $|m| = 0$:

$$\begin{array}{ll}
 & [\![R]\!] \bullet \gamma([], [\![C]\!]) \\
= & \quad \{ \text{ def. } \gamma \text{ and } \bullet \} \\
 & [] \\
= & \quad \{ \text{ def. } \gamma \text{ and } \bullet \} \\
 & \gamma([\![R]\!] \bullet [], [\![C]\!])
\end{array}$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$\begin{array}{ll}
 & [\![R]\!] \bullet \gamma(m, [\![C]\!]) \\
= & \quad \{ \text{ take } n \text{ and } o \text{ with } m = n \oplus o \wedge \exists_{\langle b,a\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))) \} \\
 & [\![R]\!] \bullet \gamma(n \oplus o, [\![C]\!]) \\
= & \quad \{ \text{ take } \langle b, a\rangle \in C \text{ with } b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{ def. } \gamma \}
\end{array}$$

$$
\begin{array}{ll}
 & [\![R]\!] \bullet ([a(\vec{d})] \oplus \gamma(o, [\![C]\!])) \\
= & \quad \{ a \in rng(C) \} \\
 & [a(\vec{d})] \oplus ([\![R]\!] \bullet \gamma(o, [\![C]\!])) \\
= & \quad \{ \text{induction hypothesis} \} \\
 & [a(\vec{d})] \oplus \gamma([\![R]\!] \bullet o, [\![C]\!]) \\
= & \quad \{ \text{def. } a \text{ and } \gamma \} \\
 & \gamma(n \oplus ([\![R]\!] \bullet o), [\![C]\!]) \\
= & \quad \{ n \in dom(C) \} \\
 & \gamma([\![R]\!] \bullet (n \oplus o), [\![C]\!]) \\
= & \quad \{ \text{def. } n \text{ and } o \} \\
 & \gamma([\![R]\!] \bullet m, [\![C]\!])
\end{array}
$$

Case $|m| > 0$ and $\exists_{n,o}(m = n \oplus o \wedge \exists_{\langle b,\tau\rangle \in C}(b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$, which is similar to the previous case.
Case $|m| > 0$ and $\neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d}))))$:

$$
\begin{array}{ll}
 & [\![R]\!] \bullet \gamma(m, [\![C]\!]) \\
= & \quad \{ |m| > 0 \Rightarrow \exists_{a}, \vec{d}, n(m = [a(\vec{d})] \oplus n), \text{ take such } a, \ \vec{d} \text{ and } n \} \\
 & [\![R]\!] \bullet \gamma([a(\vec{d})] \oplus n, [\![C]\!]) \\
= & \quad \{ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \text{ def. } \gamma \} \\
 & [\![R]\!] \bullet ([a(\vec{d})] \oplus \gamma(n, [\![C]\!])) \\
= & \quad \{ \text{def. } \bullet \} \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet \gamma(n, [\![C]\!])) \\
= & \quad \{ \text{induction hypothesis} \} \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus \gamma([\![R]\!] \bullet n, [\![C]\!]) \\
= & \quad \{ \neg\exists_{n,o}(m = n \oplus o \wedge \exists_{c \in C}((c = \langle b, a\rangle \vee c = \langle b, \tau\rangle) \wedge b = \mu(n) \wedge \exists_{\vec{d} \in \vec{D}}(\chi(n, \vec{d})))), \\
 & \quad\ \ rng(R) \cap \mathcal{N}(dom(C)) = \emptyset, \text{ def. } \gamma \} \\
 & \gamma([[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet n), [\![C]\!]) \\
= & \quad \{ \text{def. } \bullet \} \\
 & \gamma([\![R]\!] \bullet ([a(\vec{d})] \oplus n), [\![C]\!]) \\
= & \quad \{ \text{def. } a, \ \vec{d} \text{ and } n \} \\
 & \gamma([\![R]\!] \bullet m, [\![C]\!])
\end{array}
$$

C.32 *CR*2 $\Gamma_C(\rho_R(x)) \doteq \rho_R(x)$ *if* $\mathcal{N}(dom(C)) \subseteq dom(R) \wedge \mathcal{N}(dom(C)) \cap rng(R) = \emptyset$

We proof $\gamma([\![R]\!] \bullet m, [\![C]\!]) = [\![R]\!] \bullet m$ with induction on the structure of m. Case $m = []$

$$
\begin{array}{ll}
 & \gamma([\![R]\!] \bullet [], [\![C]\!]) \\
= & \quad \{ \text{def. } \bullet \text{ and } \gamma \} \\
 & [] \\
= & \quad \{ \text{def. } \bullet \} \\
 & [\![R]\!] \bullet []
\end{array}
$$

Case $m = [a(\vec{d})] \oplus n$:

$$\begin{array}{cl}
 & \gamma([\![R]\!] \bullet ([a(\vec{d})] \oplus n), [\![C]\!]) \\
= & \quad \{ \text{ def. } \textit{bullet} \ \} \\
 & \gamma([[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet n), [\![C]\!]) \\
= & \quad \{ \ [\![R]\!]^+(a) \in rng(R) \vee [\![R]\!]^+(a) \notin dom(R), \text{ def. } \textit{gamma} \ \} \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus \gamma([\![R]\!] \bullet n, [\![C]\!]) \\
= & \quad \{ \text{ induction hypothesis } \} \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet n) \\
= & \quad \{ \text{ def. } \bullet \ \} \\
 & [\![R]\!] \bullet ([a(\vec{d})] \oplus n)
\end{array}$$

C.33 *DT* $\partial_H(\tau_I(x)) \doteq \tau_I(\partial_H(x))$ *if* $[\![I]\!] \cap [\![H]\!] = \emptyset$

We proof $\mu(\theta(m, [\![I]\!])) \cap [\![H]\!] = \emptyset \equiv \mu(m) \cap [\![H']\!] = \emptyset$ with induction on the structure of m. Case $m = []$

$$\begin{array}{cl}
 & \mu(\theta([], [\![I]\!])) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \theta \ \} \\
 & \mu([]) \cap [\![H']\!] = \emptyset
\end{array}$$

Case $m = [a(\vec{d})] \oplus n$:

$$\begin{array}{cll}
 & \mu(\theta([a(\vec{d})] \oplus n, [\![I]\!])) \cap [\![H]\!] = \emptyset & \\
\equiv & \quad \{ \text{ def. } \theta \ \} & \\
 & \mu(\theta(n, [\![I]\!])) \cap [\![H]\!] = \emptyset & \textit{if } a \in [\![I]\!] \\
 & \mu([a(\vec{d})] \oplus \theta(n, [\![I]\!])) \cap [\![H]\!] = \emptyset & \textit{if } a \notin [\![I]\!] \\
\equiv & \quad \{ \text{ def. } \mu, \text{ Lemma C.1 and def. } \mathcal{N} \ \} & \\
 & \mu(\theta(n, [\![I]\!])) \cap [\![H]\!] = \emptyset & \textit{if } a \in [\![I]\!] \\
 & \mu([a(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu(\theta(n, [\![I]\!])) \cap [\![H]\!] = \emptyset & \textit{if } a \notin [\![I]\!] \\
\equiv & \quad \{ \text{ induction hypothesis } \} & \\
 & \mu(n) \cap [\![H]\!] = \emptyset & \textit{if } a \in [\![I]\!] \\
 & \mu([a(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu(n) \cap [\![H]\!] = \emptyset & \textit{if } a \notin [\![I]\!] \\
= & \quad \{ \ a \in [\![I]\!] \Rightarrow \mu([a(\vec{d})]) \cap [\![H]\!] = \emptyset \ \} & \\
 & \mu([a(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu(n) \cap [\![H]\!] = \emptyset & \textit{if } a \in [\![I]\!] \\
 & \mu([a(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu(n) \cap [\![H]\!] = \emptyset & \textit{if } a \notin [\![I]\!] \\
= & \quad \{ \text{ case elimination } \} & \\
 & \mu([a(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu(n) \cap [\![H]\!] = \emptyset & \\
= & \quad \{ \text{ def. } \mu, \text{ Lemma C.1 and def. } \mathcal{N} \ \} & \\
 & \mu([a(\vec{d})] \oplus n) \cap [\![H']\!] = \emptyset &
\end{array}$$

C.34 *DR* $\partial_H(\rho_R(x)) \doteq \rho_R(\partial_{H'}(x))$ *if* $[\![H']\!] = \{v \mid ([\![R]\!] \bullet v) \in [\![H]\!]\}$

We proof $\mu([\![R]\!] \bullet m) \cap [\![H]\!] = \emptyset \equiv \mu(m) \cap [\![H']\!] = \emptyset$ with induction on the structure of m. Case $m = []$

$$\begin{array}{cl}
 & \mu([\![R]\!] \bullet []) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \bullet, \text{ Lemma C.1 and def. } \mathcal{N} \ \}
\end{array}$$

$$
\begin{array}{ll}
 & \emptyset = \emptyset \\
\equiv & \quad \{ \text{ def. } \mathcal{N} \text{ and Lemma C.1 } \} \\
 & \mu([]) \cap [\![H']\!] = \emptyset
\end{array}
$$

Case $m = [a(\vec{d})] \oplus n$:

$$
\begin{array}{ll}
 & \mu([\![R]\!] \bullet ([a(\vec{d})] \oplus n)) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \bullet \} \\
 & \mu([[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet n)) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \bullet\text{, Lemma C.1 and def. } \mathcal{N} \} \\
 & \mu([[\![R]\!]^+(a)(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu([\![R]\!] \bullet n) \cap [\![H]\!] = \emptyset \\
\equiv & \quad \{ \text{ induction hypothesis } \} \\
 & \mu([[\![R]\!]^+(a)(\vec{d})]) \cap [\![H]\!] = \emptyset \wedge \mu(n) \cap [\![H']\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \mu \text{ and } \mathcal{N}\text{, calculus } \} \\
 & [\![R]\!]^+(a) \notin [\![H]\!] \wedge \mu(n) \cap [\![H']\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } H' \} \\
 & a \notin [\![H']\!] \wedge \mu(n) \cap [\![H']\!] = \emptyset \\
\equiv & \quad \{ \text{ def. } \mu \text{ and } \mathcal{N}\text{, calculus } \} \\
 & \mu([a(\vec{d})]) \cap [\![H']\!] = \emptyset \wedge \mu(n) \cap [\![H']\!] = \emptyset \\
\equiv & \quad \{ \text{ Lemma C.1 and def. } \mathcal{N} \} \\
 & \mu([a(\vec{d})] \oplus n) \cap [\![H']\!] = \emptyset
\end{array}
$$

C.35 *TR* $\tau_I(\rho_R(x)) \doteq \rho_R(\tau_{I'}(x))$ *if* $[\![I]\!] = \{[\![R]\!]^+(a) \mid a \in [\![I']\!]\}$

We proof $\theta([\![R]\!] \bullet m, [\![I]\!]) = [\![R]\!] \bullet \theta(m, [\![I']\!])$ with induction on the structure of m. Case $m = []$

$$
\begin{array}{ll}
 & \theta([\![R]\!] \bullet [], [\![I]\!]) \\
= & \quad \{ \text{ def. } \bullet \text{ and } \theta \} \\
 & [] \\
= & \quad \{ \text{ def. } \bullet \text{ and } \theta \} \\
 & [\![R]\!] \bullet \theta([], [\![I']\!])
\end{array}
$$

Case $m = [a(\vec{d})] \oplus n$:

$$
\begin{array}{lll}
 & \theta([\![R]\!] \bullet ([a(\vec{d})] \oplus n), [\![I]\!]) & \\
= & \quad \{ \text{ def. } \bullet \} & \\
 & \theta([[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet n), [\![I]\!]) & \\
= & \quad \{ \text{ def. } \theta \} & \\
 & \theta([\![R]\!] \bullet n, [\![I]\!]) & \textit{if } [\![R]\!]^+(a) \in [\![I]\!] \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus \theta([\![R]\!] \bullet n, [\![I]\!]) & \textit{if } [\![R]\!]^+(a) \notin [\![I]\!] \\
= & \quad \{ \text{ induction hypothesis } \} & \\
 & [\![R]\!] \bullet \theta(n, [\![I]\!]) & \textit{if } [\![R]\!]^+(a) \in [\![I]\!] \\
 & [[\![R]\!]^+(a)(\vec{d})] \oplus ([\![R]\!] \bullet \theta(n, [\![I]\!])) & \textit{if } [\![R]\!]^+(a) \notin [\![I]\!] \\
= & \quad \{ [\![R]\!]^+(a) \in [\![I]\!] \equiv a \in [\![I']\!] \text{, def, } \bullet \} &
\end{array}
$$

$$\begin{array}{lll}
 & [\![R]\!] \bullet \theta(n, [\![I]\!]) & \textit{if } a \in [\![I']\!] \\
 & [\![R]\!] \bullet ([a(\vec{d})] \oplus \theta(n, [\![I]\!])) & \textit{if } [\![R]\!]^{+}(a) \notin [\![I]\!] \\
= & \quad \{ \text{ def. } \theta,\ [\![R]\!]^{+}(a) \notin [\![I]\!] \equiv a \notin [\![I']\!] \ \} & \\
 & [\![R]\!] \bullet \theta(n, [\![I]\!]) & \textit{if } a \in [\![I']\!] \\
 & [\![R]\!] \bullet \theta([a(\vec{d})] \oplus n, [\![I]\!]) & \textit{if } a \notin [\![I']\!] \\
= & \quad \{ \text{ def. } \theta \ \} & \\
 & [\![R]\!] \bullet \theta([a(\vec{d})] \oplus n, [\![I']\!]) &
\end{array}$$

References

[1] J.C.M. Baeten, C. Verhoef, *A congruence theorem for structured operational semantics with predicates*, Computing Science Note 93/05, January 1993.

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

[3] B. Bloom, *Structural operational semantics for weak bisimulations*, Theoretical Computer Science Volume 146(1-2) pp. 25-68, 1995.

[4] J.F. Groote, S.P. Luttik, *A complete axiomatisation of branching bisimulation for process algebras with alternative quantification over data*, CWI Report SEN-R9830, November 1998.

[5] J.F. Groote, M.A. Reniers, J.J. van Wamel, M.B. van der Zwaag, *Completeness of timed μCRL*, Fundamenta Informaticae, Volume 50(3-4) pp. 361-402, April-May 2002 (also CWI Report SEN-R0034, November 2000).

[6] J.F. Groote, J. Springintveld, *Focus points and convergent process operators: a proof strategy for protocol verification*, Elsevier Journal of Logic and Algebraic Programming 49, 2001.

[7] K.M. van Hee, *Information systems engineering: a formal approach*, 1994.

[8] B. Luttik, *Choice Quantification in Process Algebra*, April 2002.

[9] W.H.J. Feijen, A.J.M. van Gasteren, *On a method of multiprogramming*, October 1999.