Theory of Computing Systems manuscript No.
(will be inserted by the editor)

A Thread Algebra with Multi-level Strategic Interleaving

J.A. Bergstra[1,2], **C.A. Middelburg**[3,1]

[1] Programming Research Group, University of Amsterdam,
P.O. Box 41882, 1009 DB Amsterdam, the Netherlands
[2] Department of Philosophy, Utrecht University,
P.O. Box 80126, 3508 TC Utrecht, the Netherlands
[3] Computing Science Department, Eindhoven University of Technology,
P.O. Box 513, 5600 MB Eindhoven, the Netherlands
e-mail: `janb@science.uva.nl, keesm@win.tue.nl`

The date of receipt and acceptance will be inserted by the editor

Abstract In a previous paper, we developed an algebraic theory about threads and a form of concurrency where some deterministic interleaving strategy determines how threads that exist concurrently are interleaved. The interleaving of different threads constitutes a multi-thread. Several multi-threads may exist concurrently on a single host in a network, several host behaviours may exist concurrently in a single network on the internet, etc. In the current paper, we assume that the above-mentioned kind of interleaving is also present at those other levels. We extend the theory developed so far with features to cover the multi-level case. We employ the resulting theory to develop a simplified, formal representation schema of the design of systems that consist of several multi-threaded programs on various hosts in different networks and to verify a property of all systems designed according to that schema.

Key words thread algebra – strategic interleaving – thread-service composition – delayed processing – exception handling – formal design prototype

1 Introduction

A thread is the behaviour of a deterministic sequential program under execution. Multi-threading refers to the concurrent existence of several threads in a program under execution. Multi-threading is the dominant form of concurrency provided by recent object-oriented programming languages such as Java [10] and C# [11].

Correspondence to: C.A. Middelburg

In the case of multi-threading, some deterministic interleaving strategy determines how threads that exist concurrently are interleaved.

Arbitrary interleaving, on which theories about concurrent processes such as ACP [5] are based, does not provide an appropriate abstraction when dealing with multi-threading: it happens that interleaving of certain threads leads to deadlock with a particular deterministic interleaving strategy whereas arbitrary interleaving would not lead to deadlock, and vice versa. In [7], we introduced a number of plausible deterministic interleaving strategies for multi-threading. We also proposed to use the phrase strategic interleaving for the more constrained form of interleaving obtained by using such a strategy. In order to deal with strategic interleaving, we assumed that a collection of threads to be interleaved takes the form of a sequence, called a thread vector.

Strategic interleaving of a thread vector constitutes a multi-thread. In conventional operating system jargon, a multi-thread is called a process. Several multi-threads may exist concurrently on the same machine. Multi-processing refers to the concurrent existence of several multi-threads on a machine. Such machines may be hosts in a network, and several host behaviours may exist concurrently in the same network. And so on and so forth. We assume that strategic interleaving is also present at those other levels.

In the current paper, we extend the theory developed so far with features to cover multi-level strategic interleaving. An axiomatic description of the features concerned, as well as a structural operational semantics, is provided. There is a dependence on the interleaving strategy considered. We extend the theory only for the simplest case, to wit cyclic interleaving. Cyclic interleaving basically operates as follows: at each stage of the interleaving, the first thread in the thread vector gets a turn to perform a step and then becomes the last one while all others move one position. Other plausible interleaving strategies are treated in [7]. They can also be adapted to the setting of multi-level strategic interleaving.

Threads proceed by performing steps, in the sequel called basic actions, in a sequential fashion. Each basic action performed by a thread is taken as a command to be processed by a service offered by the execution environment of the thread. The processing of a command may involve a change of state of the service concerned. At completion of the processing of the command, the service concerned produces a reply value which is returned to the thread. In this paper, we introduce thread-service composition, which allows for certain basic actions performed by a thread to be processed by a certain service. This is needed if certain basic actions are performed by the thread only for the sake of getting reply values returned by a certain service and that way having itself affected by that service. In such cases, the service concerned has an auxiliary nature in the sense that it forms part of the system under consideration.

We demonstrate that the theory developed in this paper may be of use by employing it to develop a simplified, formal representation schema of the design of systems that consist of several multi-threaded programs on various hosts in different networks and to verify a property of all systems designed according to that schema. We propose to use the term formal design prototype for such a schema. The verified property is laid down in a simulation theorem, which states that, if a finite thread that forms part of a system designed according to the presented schema does not make use of the services that form part of the system, then that thread is simulated by the system. In other words, the thread is not really affected by the system.

Setting up a framework in which formal design prototypes for systems that consist of several multi-threaded programs on various hosts in different networks can be developed and general properties of systems designed according to those formal design prototypes can be verified is one of the objectives with which we developed the theory presented in this paper.

The main assumption made in the theory presented in this paper is that strategic interleaving is present at all levels of such systems. This is a drastic simplification, as a result of which intuition may break down. We believe however that some such simplification is needed to obtain a manageable theory about the behaviour of such systems – and that the resulting theory will sometimes be adequate and sometimes be inadequate.

Moreover, cyclic interleaving is a simplification of the interleaving strategies actually used for multi-threading. Because of the complexity of those strategies, we consider a simplification like this one desirable to start with. It leads to an approximation which is sufficient in the case where the property laid down in the simulation theorem mentioned above is verified. The essential point turns out to be that the interleaving strategy used at each level is fair, i.e. that there will always come a next turn for all active threads, multi-threads, etc. The simulation theorem goes through for all fair interleaving strategies: the proof only depends on the use of multi-level cyclic interleaving in the part where in point of fact its fairness is shown.

Thread algebra with multi-level strategic interleaving is a design on top of BPPA (Basic Polarized Process Algebra) [6,3]. BPPA is far less general than ACP-style process algebras and its design focuses on the semantics of deterministic sequential programs. The semantics of a deterministic sequential program is supposed to be a polarized process. The idea is that a polarized process may occur in two roles: the role of a client and the role of a server. In the former role, basic actions performed by the polarized process are requests upon which a reply is expected. In the latter role, basic actions performed by the polarized process are offers to serve a request and to return a reply. The distinction between these roles is relevant in case BPPA is extended with a mechanism for client-server interaction, as in [4]. However, BPPA deals with polarized processes that occur in the role of a client only. In thread algebra, threads are regarded as polarized processes that occur in the role of a client only.

The structure of this paper is as follows. After a review of BPPA (Section 2), we extend it to a basic thread algebra with cyclic interleaving, but without any feature for multi-level strategic interleaving (Section 3). Next, we extend this basic thread algebra with thread-service composition (Section 4) and other features for multi-level strategic interleaving (Section 5). Following this, we discuss how delayed processing and exception handling can be expressed (Section 6) and give a formal representation schema of the design of systems that consist of several multi-

Table 1 Axiom of BPPA

$x \trianglelefteq \mathsf{tau} \trianglerighteq y = x \trianglelefteq \mathsf{tau} \trianglerighteq x$	T1

threaded programs on various hosts in different networks (Section 7). Finally, we make some concluding remarks (Section 8).

2 Basic Polarized Process Algebra

In this section, we review BPPA (Basic Polarized Process Algebra), a form of process algebra which is tailored to the description of the behaviour of deterministic sequential programs under execution.

In BPPA, it is assumed that there is a fixed but arbitrary finite set of *basic actions* $\mathcal{A}$ with $\mathsf{tau} \notin \mathcal{A}$. We write $\mathcal{A}_{\mathsf{tau}}$ for $\mathcal{A} \cup \{\mathsf{tau}\}$. BPPA has the following constants and operators:

- the *deadlock* constant D;
- the *termination* constant S;
- for each $a \in \mathcal{A}_{\mathsf{tau}}$, a binary *postconditional composition* operator $\_ \trianglelefteq a \trianglerighteq \_$.

We use infix notation for postconditional composition. We introduce *action prefixing* as an abbreviation: $a \circ p$, where p is a term of BPPA, abbreviates $p \trianglelefteq a \trianglerighteq p$.

The intuition is that each basic action performed by a polarized process is taken as a command to be processed by the execution environment of the polarized process. The processing of a command may involve a change of state of the execution environment. At completion of the processing of the command, the execution environment produces a reply value. This reply is either T or F and is returned to the polarized process concerned. Let p and q be closed terms of BPPA. Then $p \trianglelefteq a \trianglerighteq q$ will proceed as p if the processing of a leads to the reply T (called a positive reply), and it will proceed as q if the processing of a leads to the reply F (called a negative reply). If the reply is used to indicate whether the processing was successful, a useful convention is to indicate successful processing by the reply T and unsuccessful processing by the reply F. The action tau plays a special role. Its execution will never change any state and always produces a positive reply.

BPPA has only one axiom. This axiom is given in Table 1. Using the abbreviation introduced above, axiom T1 can be written as follows: $x \trianglelefteq \mathsf{tau} \trianglerighteq y = \mathsf{tau} \circ x$.

A *system of recursion equations* over BPPA is a set of equations $E = \{X = t_X \mid X \in V\}$ where V is a set of variables and each t_X is a term of BPPA that only contains variables from V. We write $\mathrm{V}(E)$ for the set of all variables that occur on the left-hand side of an equation in E. Let t be a term of BPPA containing a variable X. Then an occurrence of X in t is *guarded* if t has a subterm of the form $t' \trianglelefteq a \trianglerighteq t''$ containing this occurrence of X. A system of recursion equations E is *guarded* if all occurrences of variables in the right-hand sides of its equations are guarded or it can be rewritten to such a system of recursion equations using the equations of E. Following [3], a CPO structure can be imposed on the domain of the projective limit model of BPPA. Then guarded recursion equations represent

Table 2 Axioms for guarded recursion

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

continuous operators having least fixed points. These matters will not be repeated here, taking for granted that guarded systems of recursion equations have unique solutions.

We extend BPPA with guarded recursion by adding constants for solutions of guarded systems of recursion equations and axioms concerning these additional constants. For each guarded system of recursion equations E and each $X \in \mathrm{V}(E)$, we add a constant standing for the unique solution of E for X to the constants of BPPA. The constant standing for the unique solution of E for X is denoted by $\langle X|E\rangle$. Moreover, we use the following notation. Let t be a term of BPPA and E be a guarded system of recursion equations. Then we write $\langle t|E\rangle$ for t with, for all $X \in \mathrm{V}(E)$, all occurrences of X in t replaced by $\langle X|E\rangle$. We add the axioms for guarded recursion given in Table 2 to the axioms of BPPA. In this table, X, t_X and E stand for an arbitrary variable, an arbitrary term of BPPA and an arbitrary guarded system of recursion equations, respectively. Side conditions are added to restrict the variables, terms and guarded systems of recursion equations for which X, t_X and E stand. The additional axioms for guarded recursion are known as the recursive definition principle (RDP) and the recursive specification principle (RSP). The equations $\langle X|E\rangle = \langle t_X|E\rangle$ for a fixed E express that the constants $\langle X|E\rangle$ make up a solution of E. The conditional equations $E \Rightarrow X = \langle X|E\rangle$ express that this solution is the only one.

Remark 1 Let E and E' be two guarded systems of recursion equations over BPPA with $\mathrm{V}(E) = \mathrm{V}(E')$, where E' is E rewritten using the equations of E. Then, by RDP, $\langle X|E\rangle = \langle X|E'\rangle$ for all $X \in \mathrm{V}(E)$. This can be regarded as a justification of the definition of a guarded system of recursion equations. Moreover, it shows that no generality is lost if we assume in proofs that all occurrences of variables in the right-hand sides of the equations in a guarded system of recursion equations are guarded.

Henceforth, we will write BPPA(A) for BPPA with the set of basic actions $\mathcal{A}$ fixed to be the set A, and BPPA(A)+REC for BPPA(A) extended with the constants for solutions of guarded systems of recursion equations over BPPA(A) and the axioms RDP and RSP from Table 2.

The projective limit characterization of process equivalence on polarized processes is based on the notion of a finite approximation of depth n. When for all n these approximations are identical for two given polarized processes, both processes are considered identical. This is expressed by the infinitary conditional equation AIP (Approximation Induction Principle) given in Table 3. Following [6], which in fact uses the notation of [5], approximation of depth n is phrased in terms of a unary *projection* operator $\pi_n(\_)$. The projection operators are defined inductively by means of axioms P0–P3 given in Table 3. In this table and all subsequent

Table 3 Approximation induction principle

$\pi_0(x) = \mathsf{D}$	P0
$\pi_{n+1}(\mathsf{S}) = \mathsf{S}$	P1
$\pi_{n+1}(\mathsf{D}) = \mathsf{D}$	P2
$\pi_{n+1}(x \unlhd a \unrhd y) = \pi_n(x) \unlhd a \unrhd \pi_n(y)$	P3
$(\bigwedge_{n \geq 0} \pi_n(x) = \pi_n(y)) \Rightarrow x = y$	AIP

tables with axioms in which a occurs, a stands for an arbitrary action from $\mathcal{A}_{\mathsf{tau}}$. It happens that RSP follows from AIP.

Theorem 1 (RSP follows from AIP) *Let E be a guarded system of recursion equations, and let $X \in \mathrm{V}(E)$. Then it follows from* AIP *that $E \Rightarrow X = \langle X|E \rangle$.*

Proof Without loss of generality, we may assume that all occurrences of variables in the right-hand sides of the equations in E are guarded (see Remark 1). After replacing n times ($n \geq 0$) all occurrences of all $X \in \mathrm{V}(E)$ in the right-hand sides of the equations in E by the right-hand side of the equation for X in E, all occurrences of variables in the right-hand sides of the equations are at least at depth $n + 1$. We write E^n for the guarded system of recursion equations obtained in this way, and we write t_X^n for the right-hand side of the equation for X in E^n. Because all occurrences of variables in t_X^n are at least at depth $n + 1$, $\pi_n(t_X^n)$ is a closed term. Now assume E and take an arbitrary $n \geq 0$. Then E^n and in particular $X = t_X^n$. From this, it follows immediately that $\pi_n(X) = \pi_n(t_X^n)$. Hence, $E \Rightarrow \pi_n(X) = \pi_n(t_X^n)$. From this, and the fact that $\pi_n(t_X^n)$ equals a closed term, it follows by RDP that also $\pi_n(\langle X|E \rangle) = \pi_n(t_X^n)$. Hence, $\pi_n(X) = \pi_n(\langle X|E \rangle)$. From this, it follows by AIP that $X = \langle X|E \rangle$. $\square$

As mentioned above, the behaviour of a polarized process depends upon its execution environment. Each basic action performed by the polarized process is taken as a command to be processed by the execution environment. At any stage, the commands that the execution environment can accept depend only on its history, i.e. the sequence of commands processed before and the sequence of replies produced for those commands. When the execution environment accepts a command, it will produce a positive reply or a negative reply. Whether the reply is positive or negative usually depends on the execution history. However, it may also depend on external conditions.

In the structural operational semantics, we represent an execution environment by a function $\rho : (\mathcal{A} \times \{\mathsf{T}, \mathsf{F}\})^* \to \mathcal{P}(\mathcal{A} \times \{\mathsf{T}, \mathsf{F}\})$ that satisfies the following condition: $(a, b) \notin \rho(\alpha) \Rightarrow \rho(\alpha \frown \langle (a, b) \rangle) = \emptyset$ for all $a \in \mathcal{A}$, $b \in \{\mathsf{T}, \mathsf{F}\}$ and $\alpha \in (\mathcal{A} \times \{\mathsf{T}, \mathsf{F}\})^*$.[1] We write $\mathcal{E}$ for the set of all those functions. Given an execution environment $\rho \in \mathcal{E}$ and a basic action $a \in \mathcal{A}$, the *derived* execution environment of ρ after processing a with a *positive* reply, written $\frac{\partial^+}{\partial a}\rho$, is

[1] We write $\langle\,\rangle$ for the empty sequence, $\langle d \rangle$ for the sequence having d as sole element, and $\alpha \frown \beta$ for the concatenation of sequences α and β. We assume that the identities $\alpha \frown \langle\,\rangle = \langle\,\rangle \frown \alpha = \alpha$ hold.

Table 4 Transition rules of BPPA

$$\frac{}{\mathsf{S}\downarrow} \qquad \frac{}{\mathsf{D}\uparrow} \qquad \frac{}{\langle x \unlhd \mathsf{tau} \unrhd y, \rho\rangle \xrightarrow{\mathsf{tau}} \langle x, \rho\rangle}$$

$$\frac{}{\langle x \unlhd a \unrhd y, \rho\rangle \xrightarrow{a} \langle x, \frac{\partial^+}{\partial a}\,\rho\rangle}\ (a, \mathsf{T}) \in \rho(\langle\,\rangle) \qquad \frac{}{\langle x \unlhd a \unrhd y, \rho\rangle \xrightarrow{a} \langle y, \frac{\partial^-}{\partial a}\,\rho\rangle}\ (a, \mathsf{F}) \in \rho(\langle\,\rangle)$$

$$\frac{x\downarrow}{x\updownarrow} \qquad \frac{x\uparrow}{x\updownarrow}$$

Table 5 Transition rules for guarded recursion

$$\frac{\langle\langle t|E\rangle, \rho\rangle \xrightarrow{a} \langle x', \rho'\rangle}{\langle\langle X|E\rangle, \rho\rangle \xrightarrow{a} \langle x', \rho'\rangle}\ X{=}t \in E \qquad \frac{\langle t|E\rangle\downarrow}{\langle X|E\rangle\downarrow}\ X{=}t \in E \qquad \frac{\langle t|E\rangle\uparrow}{\langle X|E\rangle\uparrow}\ X{=}t \in E$$

defined by $\frac{\partial^+}{\partial a}\,\rho(\alpha) = \rho(\langle(a, \mathsf{T})\rangle \frown \alpha)$; and likewise the *derived* execution environment of ρ after processing a with a *negative* reply, written $\frac{\partial^-}{\partial a}\,\rho$, is defined by $\frac{\partial^-}{\partial a}\,\rho(\alpha) = \rho(\langle(a, \mathsf{F})\rangle \frown \alpha)$.

The following transition relations on closed terms of BPPA are used in the structural operational semantics of BPPA:

- a binary relation $\langle \_, \rho\rangle \xrightarrow{a} \langle \_, \rho'\rangle$ for each $a \in \mathcal{A}_{\mathsf{tau}}$ and $\rho, \rho' \in \mathcal{E}$;
- a unary relation $\_\downarrow$;
- a unary relation $\_\uparrow$;
- a unary relation $\_\updownarrow$.

The four kinds of transition relations are called the *action step*, *termination*, *deadlock*, and *termination or deadlock* relations, respectively. They can be explained as follows:

- $\langle p, \rho\rangle \xrightarrow{a} \langle p', \rho'\rangle$: in execution environment ρ, process p can perform action a and after that proceed as process p' in execution environment ρ';
- $p\downarrow$: process p cannot but terminate successfully;
- $p\uparrow$: process p cannot but become inactive;
- $p\updownarrow$: process p cannot but terminate successfully or become inactive.

The termination or deadlock relation is an auxiliary relation needed when we extend BPPA in Section 3.

The structural operational semantics of BPPA is described by the transition rules given in Table 4. In this table and all subsequent tables with transition rules in which a occurs, a stands for an arbitrary action from $\mathcal{A}_{\mathsf{tau}}$. The transition rules for the constants for solutions of guarded systems of recursion equations over BPPA are given in Table 5. In this table, X, t_X and E stand for an arbitrary variable, an arbitrary term of BPPA and an arbitrary guarded system of recursion equations over BPPA, respectively. The transition rules for projection are given in Table 6.

Bisimulation equivalence is defined as follows. A *bisimulation* is a symmetric binary relation B on closed terms of BPPA such that for all closed terms p and q:

Table 6 Transition rules for projection

$$\frac{\langle x, \rho\rangle \xrightarrow{a} \langle x', \rho'\rangle}{\langle \pi_{n+1}(x), \rho\rangle \xrightarrow{a} \langle \pi_n(x'), \rho'\rangle} \qquad \frac{x\downarrow}{\pi_{n+1}(x)\downarrow} \qquad \frac{x\uparrow}{\pi_{n+1}(x)\uparrow} \qquad \frac{}{\pi_0(x)\uparrow}$$

– if $B(p, q)$ and $\langle p, \rho\rangle \xrightarrow{a} \langle p', \rho'\rangle$, then there is a q' such that $\langle q, \rho\rangle \xrightarrow{a} \langle q', \rho'\rangle$ and $B(p', q')$;
– if $B(p, q)$ and $p\downarrow$, then $q\downarrow$;
– if $B(p, q)$ and $p\uparrow$, then $q\uparrow$.

Two closed terms p and q are *bisimulation equivalent*, written $p \leftrightarrows q$, if there exists a bisimulation B such that $B(p, q)$.

Bisimulation equivalence is a congruence with respect to the postconditional composition operators and the projection operators. This follows immediately from the fact that the transition rules for these operators are in the path format (see e.g. [2]). The axioms given in Tables 1, 2 and 3 are sound with respect to bisimulation equivalence.

3 A Basic Thread Algebra with Foci and Methods

In this section, we introduce a thread algebra that deals with single-level strategic interleaving. Features for multi-level strategic interleaving will be added in subsequent sections. The thread algebra introduced in this section is an extension of BPPA. In [6], its has been outlined how and why polarized processes are a natural candidate for the specification of the semantics of deterministic sequential programs. Assuming that a thread is a process representing a deterministic sequential program under execution, it is reasonable to view all polarized processes as threads.

In order to deal with strategic interleaving, it is assumed that a collection of threads to be interleaved takes the form of a sequence, called a thread vector. Strategic interleaving operators turn a thread vector of arbitrary length into a single thread. This single thread obtained via a strategic interleaving operator is also called a multi-thread. Formally, however both threads and multi-threads are polarized processes.

In this paper, we only cover the simplest interleaving strategy, namely *cyclic interleaving*. Cyclic interleaving basically operates as follows: at each stage of the interleaving, the first thread in the thread vector gets a turn to perform a basic action and then the thread vector undergoes cyclic permutation. We mean by cyclic permutation of a thread vector that the first thread in the thread vector becomes the last one and all others move one position to the left. If one thread in the thread vector deadlocks, the whole does not deadlock till all others have terminated or deadlocked. An important property of cyclic interleaving is that it is fair, i.e. there will always come a next turn for all active threads.

Other plausible interleaving strategies are treated in [7]. They can also be adapted to the features for multi-level level strategic interleaving that will be in-

Table 7 Axioms for cyclic interleaving

$\|(\langle\,\rangle) = \mathsf{S}$	CSI1
$\|(\langle \mathsf{S}\rangle \frown \alpha) = \|(\alpha)$	CSI2
$\|(\langle \mathsf{D}\rangle \frown \alpha) = \mathsf{S}_\mathsf{D}(\|(\alpha))$	CSI3
$\|(\langle \mathsf{tau} \circ x\rangle \frown \alpha) = \mathsf{tau} \circ \|(\alpha \frown \langle x\rangle)$	CSI4
$\|(\langle x \trianglelefteq f.m \trianglerighteq y\rangle \frown \alpha) = \|(\alpha \frown \langle x\rangle) \trianglelefteq f.m \trianglerighteq \|(\alpha \frown \langle y\rangle)$	CSI5

Table 8 Axioms for deadlock at termination

$\mathsf{S}_\mathsf{D}(\mathsf{S}) = \mathsf{D}$	S2D1
$\mathsf{S}_\mathsf{D}(\mathsf{D}) = \mathsf{D}$	S2D2
$\mathsf{S}_\mathsf{D}(\mathsf{tau} \circ x) = \mathsf{tau} \circ \mathsf{S}_\mathsf{D}(x)$	S2D3
$\mathsf{S}_\mathsf{D}(x \trianglelefteq f.m \trianglerighteq y) = \mathsf{S}_\mathsf{D}(x) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(y)$	S2D4

troduced in the current paper. The strategic interleaving operator for cyclic interleaving is denoted by $\|(\_)$. In [7], it was denoted by $\|_{csi}(\_)$ to distinguish it from other strategic interleaving operators.

It is assumed that there is a fixed but arbitrary finite set of *foci* $\mathcal{F}$ and a fixed but arbitrary finite set of *methods* $\mathcal{M}$. For the set of basic actions $\mathcal{A}$, we take the set $FM = \{f.m \mid f \in \mathcal{F}, m \in \mathcal{M}\}$. Each focus plays the role of a name of a service provided by the execution environment that can be requested to process a command. Each method plays the role of a command proper. Performing a basic action $f.m$ is taken as making a request to the service named f to process the command m.

The axioms for cyclic interleaving are given in Table 7. In this table and all subsequent tables with axioms or transition rules in which f and m occur, f and m stand for an arbitrary focus from $\mathcal{F}$ and an arbitrary method from $\mathcal{M}$, respectively. In CSI3, the auxiliary *deadlock at termination* operator $\mathsf{S}_\mathsf{D}(\_)$ is used. This operator turns termination into deadlock. Its axioms appear in Table 8.

Henceforth, we will write $\mathrm{TA}_{\mathrm{fm}}$ for BPPA(FM) extended with the strategic interleaving operator for cyclic interleaving, the deadlock at termination operator, and the axioms from Tables 7 and 8.

Example 1 The following equation is easily derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}$:

$$
\begin{aligned}
&\|(\langle (f_1'.m_1' \circ \mathsf{S}) \trianglelefteq f_1.m_1 \trianglerighteq (f_1''.m_1'' \circ \mathsf{S})\rangle \frown \\
&\quad \langle (f_2'.m_2' \circ \mathsf{S}) \trianglelefteq f_2.m_2 \trianglerighteq (f_2''.m_2'' \circ \mathsf{S})\rangle) \\
&= ((f_1'.m_1' \circ f_2'.m_2' \circ \mathsf{S}) \trianglelefteq f_2.m_2 \trianglerighteq (f_1'.m_1' \circ f_2''.m_2'' \circ \mathsf{S})) \\
&\quad \trianglelefteq f_1.m_1 \trianglerighteq \\
&\quad ((f_1''.m_1'' \circ f_2'.m_2' \circ \mathsf{S}) \trianglelefteq f_2.m_2 \trianglerighteq (f_1''.m_1'' \circ f_2''.m_2'' \circ \mathsf{S}))\;.
\end{aligned}
$$

This equation shows clearly that the two threads $(f_1'.m_1' \circ \mathsf{S}) \trianglelefteq f_1.m_1 \trianglerighteq (f_1''.m_1'' \circ \mathsf{S})$ and $(f_2'.m_2' \circ \mathsf{S}) \trianglelefteq f_2.m_2 \trianglerighteq (f_2''.m_2'' \circ \mathsf{S})$ are interleaved in a cyclic manner: first the first thread performs $f_1.m_1$, next the second thread performs $f_2.m_2$, next the first thread performs $f'_1.m'_1$ or $f''_1.m''_1$ depending upon the reply on $f_1.m_1$, next the second thread performs $f'_2.m'_2$ or $f''_2.m''_2$ depending upon the reply on $f_2.m_2$.

We can prove that each closed term of TA$_{\mathrm{fm}}$ can be reduced to a closed term of BPPA(FM).

Theorem 2 (Elimination) *For all closed terms p of* TA$_{\mathrm{fm}}$*, there exists a closed term q of* BPPA(FM) *such that $p = q$ is derivable from the axioms of* TA$_{\mathrm{fm}}$.

Proof We prove this by induction on the structure of p:

- $p \equiv \mathsf{S}$: S is a closed term of BPPA(FM).
- $p \equiv \mathsf{D}$: D is a closed term of BPPA(FM).
- $p \equiv \mathtt{tau} \circ p'$: Let q' be a closed term of BPPA(FM) such that $p' = q'$. Such a term exists by the induction hypothesis. Then $\mathtt{tau} \circ q'$ is a closed term of BPPA(FM) and $\mathtt{tau} \circ p' = \mathtt{tau} \circ q'$.
- $p \equiv p' \trianglelefteq f.m \trianglerighteq p''$: Let q' and q'' be closed terms of BPPA(FM) such that $p' = q'$ and $p'' = q''$. Such terms exist by the induction hypothesis. Then $q' \trianglelefteq f.m \trianglerighteq q''$ is a closed term of BPPA(FM) and $p' \trianglelefteq f.m \trianglerighteq p'' = q' \trianglelefteq f.m \trianglerighteq q''$.
- $p \equiv \mathsf{S}_\mathsf{D}(p')$: By the induction hypothesis, there exists a closed term q' of BPPA(FM) such that $p' = q'$. So we are done if we have proved the following lemma:

  Let q' be a closed term of BPPA(FM). Then there exists a closed term r' of BPPA(FM) such that $\mathsf{S}_\mathsf{D}(q') = r'$ is derivable from the axioms of TA$_{\mathrm{fm}}$.

  We prove this lemma by induction on the structure of q':
  - $q' \equiv \mathsf{S}$: $\mathsf{S}_\mathsf{D}(\mathsf{S}) = \mathsf{D}$ by S2D1 and D is a closed term of BPPA(FM).
  - $q' \equiv \mathsf{D}$: $\mathsf{S}_\mathsf{D}(\mathsf{D}) = \mathsf{D}$ by S2D2 and D is a closed term of BPPA(FM).
  - $q' \equiv \mathtt{tau} \circ q''$: $\mathsf{S}_\mathsf{D}(\mathtt{tau} \circ q'') = \mathtt{tau} \circ \mathsf{S}_\mathsf{D}(q'')$ by S2D3. Let r'' be a closed term of BPPA(FM) such that $\mathsf{S}_\mathsf{D}(q'') = r''$. Such a term exists by the induction hypothesis. Then $\mathtt{tau} \circ r''$ is a closed term of BPPA(FM) and $\mathsf{S}_\mathsf{D}(\mathtt{tau} \circ q'') = \mathtt{tau} \circ r''$.
  - $q' \equiv q'' \trianglelefteq f.m \trianglerighteq q'''$: $\mathsf{S}_\mathsf{D}(q'' \trianglelefteq f.m \trianglerighteq q''') = \mathsf{S}_\mathsf{D}(q'') \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(q''')$ by S2D4. Let r'' and r''' be closed terms of BPPA(FM) such that $\mathsf{S}_\mathsf{D}(q'') = r''$ and $\mathsf{S}_\mathsf{D}(q''') = r'''$. Such terms exist by the induction hypothesis. Then $r'' \trianglelefteq f.m \trianglerighteq r'''$ is a closed term of BPPA(FM) and $\mathsf{S}_\mathsf{D}(q'' \trianglelefteq f.m \trianglerighteq q''') = r'' \trianglelefteq f.m \trianglerighteq r'''$.
- $p \equiv \|(\alpha)$: If $\alpha = \langle\,\rangle$, then $\|(\alpha) = \mathsf{S}$ by CSI1 and S is a closed term of BPPA(FM). If $\alpha = \|(\langle p'_1\rangle \frown \ldots \frown \langle p'_n\rangle)$ for some $n > 0$, then, by the induction hypothesis, there exist closed terms $q'_1, \ldots, q'_n$ of BPPA(FM) such that $p'_1 = q'_1, \ldots, p'_n = q'_n$. So we are done if we have proved the following lemma:

  Let $q'_1, \ldots, q'_n$ ($n > 0$) be closed terms of BPPA(FM). Then there exists a closed term r' of BPPA(FM) such that $\|(\langle q'_1\rangle \frown \ldots \frown \langle q'_n\rangle) = r'$ is derivable from the axioms of TA$_{\mathrm{fm}}$.

  We prove this lemma by induction on the sum of the depths plus one of $q'_1, \ldots, q'_n$ and case distinction on the structure of q'_1:

- $q'_1 \equiv \mathsf{S}$: $\|(\langle \mathsf{S}\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = \|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle)$ by CSI2. Let r' be a closed term of BPPA(FM) such that $\|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = r'$. Such a term exists by the induction hypothesis. Moreover, $\|(\langle \mathsf{S}\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = r'$.
- $q'_1 \equiv \mathsf{D}$: $\|(\langle \mathsf{D}\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = \mathsf{S}_\mathsf{D}(\|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle))$ by CSI3. Let r' be a closed term of BPPA(FM) such that $\|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = r'$. Such a term exists by the induction hypothesis. Let s' be a closed term of BPPA(FM) such that $\mathsf{S}_\mathsf{D}(r') = s'$. Such a term exists by the lemma proved above for the case $p \equiv \mathsf{S}_\mathsf{D}(p')$. Moreover, $\|(\langle \mathsf{D}\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = s'$.
- $q'_1 \equiv \mathsf{tau} \circ q''_1$: $\|(\langle \mathsf{tau} \circ q''_1\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = \mathsf{tau} \circ \|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle \curvearrowright \langle q''_1\rangle)$ by CSI4. Let r' be a closed term of BPPA(FM) such that $\|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle \curvearrowright \langle q''_1\rangle) = r'$. Such a term exists by the induction hypothesis. Then $\mathsf{tau} \circ r'$ is a closed term of BPPA(FM) and $\|(\langle \mathsf{tau} \circ q''_1\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = \mathsf{tau} \circ r'$.
- $q'_1 \equiv q''_1 \trianglelefteq f.m \trianglerighteq q'''_1$: $\|(\langle q''_1 \trianglelefteq f.m \trianglerighteq q'''_1\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = \|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle \curvearrowright \langle q''_1\rangle) \trianglelefteq f.m \trianglerighteq \|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle \curvearrowright \langle q'''_1\rangle)$ by CSI5. Let r' and r'' be closed terms of BPPA(FM) such that $\|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle \curvearrowright \langle q''_1\rangle) = r'$ and $\|(\langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle \curvearrowright \langle q'''_1\rangle) = r''$. Such terms exist by the induction hypothesis. Then $r' \trianglelefteq f.m \trianglerighteq r''$ is a closed term of BPPA(FM) and $\|(\langle q''_1 \trianglelefteq f.m \trianglerighteq q'''_1\rangle \curvearrowright \langle q'_2\rangle \curvearrowright \ldots \curvearrowright \langle q'_n\rangle) = r' \trianglelefteq f.m \trianglerighteq r''$.

□

The following proposition, concerning the cyclic interleaving of a thread vector of length 1, is easily proved using Theorem 2.

Proposition 1 *For all closed terms p of* $\mathrm{TA}_{\mathrm{fm}}$*, the equation $\|(\langle p\rangle) = p$ is derivable from the axioms of* $\mathrm{TA}_{\mathrm{fm}}$.

Proof By Theorem 2, it is sufficient to prove that this equation is derivable for all closed terms p of BPPA(FM). We prove this by induction on the structure of p:

- $p \equiv \mathsf{S}$: $\|(\langle \mathsf{S}\rangle) = \mathsf{S}$ by CSI2 and CSI1.
- $p \equiv \mathsf{D}$: $\|(\langle \mathsf{D}\rangle) = \mathsf{D}$ by CSI3, CSI1 and S2D1.
- $p \equiv \mathsf{tau} \circ p'$: $\|(\langle \mathsf{tau} \circ p'\rangle) = \mathsf{tau} \circ p'$ by CSI4 and the induction hypothesis.
- $p \equiv p' \trianglelefteq f.m \trianglerighteq p''$: $\|(\langle p' \trianglelefteq f.m \trianglerighteq p''\rangle) = p' \trianglelefteq f.m \trianglerighteq p''$ by CSI5 and the induction hypothesis.

In the proof of each case, in addition to the above-mentioned axioms, the fact that $\alpha = \alpha \curvearrowright \langle\,\rangle = \langle\,\rangle \curvearrowright \alpha$ is needed. □

The equation $\|(\langle p\rangle) = p$ from Proposition 1 expresses the obvious fact that in the cyclic interleaving of a thread vector of length 1 no proper interleaving is involved.

The following are useful properties of the deadlock at termination operator which are proved using Theorem 2 as well.

Proposition 2 *For all closed terms $p_1, \ldots, p_n$ of* $\mathrm{TA}_{\mathrm{fm}}$*, the following equations are derivable from the axioms of* $\mathrm{TA}_{\mathrm{fm}}$*:*

$$\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p_1)) = \mathsf{S}_\mathsf{D}(p_1)\ , \tag{1}$$

$$\mathsf{S}_\mathsf{D}(\|(\langle p_1\rangle \curvearrowright \ldots \curvearrowright \langle p_n\rangle)) = \|(\langle \mathsf{S}_\mathsf{D}(p_1)\rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n)\rangle)\ . \tag{2}$$

Proof By Theorem 2, it is sufficient to prove that these equations are derivable for all closed terms $p_1, \ldots, p_n$ of BPPA(FM). We prove that (1) is derivable by induction on the structure of p_1:

– $p_1 \equiv \mathsf{S}$: $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\mathsf{S})) = \mathsf{D}$ by S2D1 and S2D2, and $\mathsf{D} = \mathsf{S}_\mathsf{D}(\mathsf{S})$ by S2D1.
– $p_1 \equiv \mathsf{D}$: $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\mathsf{D})) = \mathsf{S}_\mathsf{D}(\mathsf{D})$ by S2D2.
– $p_1 \equiv \mathsf{tau} \circ p'_1$: $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\mathsf{tau} \circ p'_1)) = \mathsf{tau} \circ \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p'_1))$ by S2D3 twice, $\mathsf{tau} \circ \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p'_1)) = \mathsf{tau} \circ \mathsf{S}_\mathsf{D}(p'_1)$ by the induction hypothesis, and $\mathsf{tau} \circ \mathsf{S}_\mathsf{D}(p'_1) = \mathsf{S}_\mathsf{D}(\mathsf{tau} \circ p'_1)$ by S2D3.
– $p_1 \equiv p'_1 \trianglelefteq f.m \trianglerighteq p''_1$: $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p'_1 \trianglelefteq f.m \trianglerighteq p''_1)) = \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p'_1)) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p''_1))$ by S2D4 twice, $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p'_1)) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(p''_1)) = \mathsf{S}_\mathsf{D}(p'_1) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(p''_1)$ by the induction hypothesis, and $\mathsf{S}_\mathsf{D}(p'_1) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(p''_1) = \mathsf{S}_\mathsf{D}(p'_1 \trianglelefteq f.m \trianglerighteq p''_1)$ by S2D4.

We prove that (2) is derivable by induction on the sum of the depths plus one of $p_1, \ldots, p_n$ and case distinction on the structure of p_1:

– $p_1 \equiv \mathsf{S}$: $\mathsf{S}_\mathsf{D}(\|(\langle \mathsf{S} \rangle \curvearrowright \langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)) = \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)))$ by CSI2 and (1), $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle))) = \mathsf{S}_\mathsf{D}(\|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle))$ by the induction hypothesis, and $\mathsf{S}_\mathsf{D}(\|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle)) = \|(\langle \mathsf{S}_\mathsf{D}(\mathsf{S}) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle)$ by CSI3 and S2D1.
– $p_1 \equiv \mathsf{D}$: $\mathsf{S}_\mathsf{D}(\|(\langle \mathsf{D} \rangle \curvearrowright \langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)) = \mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)))$ by CSI3, $\mathsf{S}_\mathsf{D}(\mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle))) = \mathsf{S}_\mathsf{D}(\|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle))$ by the induction hypothesis, and $\mathsf{S}_\mathsf{D}(\|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle)) = \|(\langle \mathsf{S}_\mathsf{D}(\mathsf{D}) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle)$ by CSI3 and S2D2.
– $p_1 \equiv \mathsf{tau} \circ p'_1$: $\mathsf{S}_\mathsf{D}(\|(\langle \mathsf{tau} \circ p'_1 \rangle \curvearrowright \langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)) = \mathsf{tau} \circ \mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle \curvearrowright \langle p'_1 \rangle))$ by CSI4 and S2D3, $\mathsf{tau} \circ \mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle \curvearrowright \langle p'_1 \rangle)) = \mathsf{tau} \circ \|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p'_1) \rangle)$ by the induction hypothesis, and $\mathsf{tau} \circ \|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p'_1) \rangle) = \|(\langle \mathsf{S}_\mathsf{D}(\mathsf{tau} \circ p'_1) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle)$ by CSI4 and S2D3.
– $p_1 \equiv p'_1 \trianglelefteq f.m \trianglerighteq p''_1$: $\mathsf{S}_\mathsf{D}(\|(\langle p'_1 \trianglelefteq f.m \trianglerighteq p''_1 \rangle \curvearrowright \langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)) = \mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle \curvearrowright \langle p'_1 \rangle)) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle \curvearrowright \langle p''_1 \rangle))$ by CSI5 and S2D4, $\mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle \curvearrowright \langle p'_1 \rangle)) \trianglelefteq f.m \trianglerighteq \mathsf{S}_\mathsf{D}(\|(\langle p_2 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle \curvearrowright \langle p''_1 \rangle)) = \|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p'_1) \rangle) \trianglelefteq f.m \trianglerighteq \|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p''_1) \rangle)$ by the induction hypothesis, and $\|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p'_1) \rangle) \trianglelefteq f.m \trianglerighteq \|(\langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p''_1) \rangle) = \|(\langle \mathsf{S}_\mathsf{D}(p'_1 \trianglelefteq f.m \trianglerighteq p''_1) \rangle \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_2) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_\mathsf{D}(p_n) \rangle)$ by CSI5 and S2D4.

□

We extend TA$_\mathrm{fm}$ with guarded recursion like in the case of BPPA. It involves systems of recursion equations over TA$_\mathrm{fm}$, which require an adaptation of the notion of guardedness. A *system of recursion equations* over TA$_\mathrm{fm}$ is a set of equations $E = \{X = t_X \mid X \in V\}$ where V is a set of variables and each t_X is a term of TA$_\mathrm{fm}$ that only contains variables from V. Let t be a term of TA$_\mathrm{fm}$ containing a variable X. Then an occurrence of X in t is *guarded* if t has a subterm of the form $t' \trianglelefteq a \trianglerighteq t''$ containing this occurrence of X. A system of recursion equations E is *guarded* if all occurrences of variables in the right-hand sides of its equations are guarded or it can be rewritten to such a system of recursion equations using the axioms of $\mathrm{TA}_{\mathrm{fm}}$ and the equations of E.

Henceforth, we will write $\mathrm{TA}_{\mathrm{fm}}$+REC for $\mathrm{TA}_{\mathrm{fm}}$ extended with the constants for solutions of guarded systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}$ and the axioms RDP and RSP from Table 2.

Theorem 2 states that the strategic interleaving operator for cyclic interleaving and the deadlock at termination operator can be eliminated from closed terms of $\mathrm{TA}_{\mathrm{fm}}$. It does not state anything concerning closed terms of $\mathrm{TA}_{\mathrm{fm}}$+REC. The following two propositions concern the case where the operand of the strategic interleaving operator for cyclic interleaving is a sequence of constants for solutions of guarded systems of recursion equations over BPPA(FM) and the case where the operand of the deadlock at termination operator is such a constant.

Proposition 3 *Let E' and E'' be guarded systems of recursion equations over* BPPA(FM), *let $X \in \mathrm{V}(E')$, and let $Y \in \mathrm{V}(E'')$. Then there exists a guarded system of recursion equations E over* BPPA(FM) *and a variable $Z \in \mathrm{V}(E)$ such that $\|(\langle\langle X|E'\rangle\rangle \curvearrowright \langle\langle Y|E''\rangle\rangle) = \langle Z|E\rangle$ is derivable from the axioms of* $\mathrm{TA}_{\mathrm{fm}}$+REC.

Proof Without loss of generality, we may assume that all occurrences of variables in the right-hand sides of the equations in E' and E'' are guarded (see Remark 1). Without loss of generality, we may also assume that $\mathrm{V}(E')$ and $\mathrm{V}(E'')$ are disjoint sets. We take an injective function Z that maps each pair of variables in $(\mathrm{V}(E') \times \mathrm{V}(E'')) \cup (\mathrm{V}(E'') \times \mathrm{V}(E'))$ to a variable not in $\mathrm{V}(E') \cup \mathrm{V}(E'')$, and define, guided by axioms CSI2–CSI5, the following guarded system of recursion equations:

$$
\begin{aligned}
E = {} & \{Z(X',Y') = Y' \mid X' = \mathsf{S} \in E' \wedge Y' \in \mathrm{V}(E'')\} \\
& \cup \{Z(X',Y') = \mathsf{S}_{\mathsf{D}}(Y') \mid X' = \mathsf{D} \in E' \wedge Y' \in \mathrm{V}(E'')\} \\
& \cup \{Z(X',Y') = \mathsf{tau} \circ Z(Y',X'') \mid \\
& \qquad X' = \mathsf{tau} \circ X'' \in E' \wedge Y' \in \mathrm{V}(E'')\} \\
& \cup \{Z(X',Y') = Z(Y',X'') \trianglelefteq f.m \trianglerighteq Z(Y',X''') \mid \\
& \qquad X' = X'' \trianglelefteq f.m \trianglerighteq X''' \in E' \wedge Y' \in \mathrm{V}(E'')\} \\
& \cup \{Z(Y',X') = X' \mid Y' = \mathsf{S} \in E'' \wedge X' \in \mathrm{V}(E')\} \\
& \cup \{Z(Y',X') = \mathsf{S}_{\mathsf{D}}(X') \mid Y' = \mathsf{D} \in E'' \wedge X' \in \mathrm{V}(E')\} \\
& \cup \{Z(Y',X') = \mathsf{tau} \circ Z(X',Y'') \mid \\
& \qquad Y' = \mathsf{tau} \circ Y'' \in E'' \wedge X' \in \mathrm{V}(E')\} \\
& \cup \{Z(Y',X') = Z(X',Y'') \trianglelefteq f.m \trianglerighteq Z(X',Y''') \mid \\
& \qquad Y' = Y'' \trianglelefteq f.m \trianglerighteq Y''' \in E'' \wedge X' \in \mathrm{V}(E')\} \\
& \cup E' \cup E'' \,.
\end{aligned}
$$

If we replace in E, for all $X' \in \mathrm{V}(E')$ and all $Y' \in \mathrm{V}(E'')$, all occurrences of $Z(X',Y')$ by $\|(\langle\langle X'|E'\rangle\rangle \curvearrowright \langle\langle Y'|E''\rangle\rangle)$, all occurrences of $Z(Y',X')$ by $\|(\langle\langle Y'|E''\rangle\rangle \curvearrowright \langle\langle X'|E'\rangle\rangle)$, all occurrences of X' by $\|(\langle\langle X'|E'\rangle\rangle)$ and all occurrences of Y' by $\|(\langle\langle Y'|E''\rangle\rangle)$, then each of the resulting equations is derivable by first applying RDP and then applying one of CSI2–CSI5. Hence, $\|(\langle\langle X|E'\rangle\rangle \curvearrowright$

$\langle\langle Y|E''\rangle\rangle)$ is a solution of E for $Z(X,Y)$. From this, it follows by RSP that $\|(\langle\langle X|E'\rangle\rangle \frown \langle\langle Y|E''\rangle\rangle) = \langle Z(X,Y)|E\rangle$. $\square$

Proposition 4 *Let E' be a guarded system of recursion equations over* BPPA(FM)*, and let $X \in \mathrm{V}(E')$. Then there exists a guarded system of recursion equations E over* BPPA(FM) *and a variable $Y \in \mathrm{V}(E)$ such that* $\mathsf{S_D}(\langle X|E'\rangle) = \langle Y|E\rangle$ *is derivable from the axioms of* TA$_{\mathrm{fm}}$+REC.

Proof Without loss of generality, we may assume that all occurrences of variables in the right-hand sides of the equations in E' are guarded. We take an injective function Y that maps each variable in $\mathrm{V}(E')$ to a variable not in $\mathrm{V}(E')$, and define, guided by axioms S2D1–S2D4, the following guarded system of recursion equations:

$$
\begin{aligned}
E = {} & \{Y(X') = \mathsf{D} \mid X' = \mathsf{S} \in E'\} \cup \{Y(X') = \mathsf{D} \mid X' = \mathsf{D} \in E'\} \\
& \cup \{Y(X') = \mathsf{tau} \circ Y(X'') \mid X' = \mathsf{tau} \circ X'' \in E'\} \\
& \cup \{Y(X') = Y(X'') \trianglelefteq f.m \trianglerighteq Y(X''') \mid X' = X'' \trianglelefteq f.m \trianglerighteq X''' \in E'\} \; .
\end{aligned}
$$

If we replace in E, for all $X' \in \mathrm{V}(E')$, all occurrences of $Y(X')$ by $\mathsf{S_D}(\langle X'|E'\rangle)$, then each of the resulting equations is derivable by first applying RDP and then applying one of S2D1–S2D4. Hence, $\mathsf{S_D}(\langle X|E'\rangle)$ is a solution of E for $Y(X)$. From this, it follows by RSP that $\mathsf{S_D}(\langle X|E'\rangle) = \langle Y(X)|E\rangle$. $\square$

Proposition 3 states that the strategic interleaving operator for cyclic interleaving can be eliminated from terms of the form $\|(\langle\langle X|E'\rangle\rangle \frown \langle\langle Y|E''\rangle\rangle)$ if E' and E'' are guarded systems of recursion equations over BPPA(FM). Proposition 4 states that the deadlock at termination operator can be eliminated from terms of the form $\mathsf{S_D}(\langle X|E'\rangle)$ if E' is a guarded system of recursion equations over BPPA(FM). Moreover, both state that the resulting term is a term of the form $\langle Z|E\rangle$ where E is a guarded system of recursion equations over BPPA(FM). It is clear that the proof of Proposition 3 generalizes to the case where the operand is a sequence of length greater than 2.

The structural operational semantics of TA$_{\mathrm{fm}}$ is described by the transition rules given in Tables 4 and 9.

Bisimulation equivalence is also a congruence with respect to the strategic interleaving operator for cyclic interleaving and the deadlock at termination operator. This follows immediately from the fact that the transition rules for TA$_{\mathrm{fm}}$ constitute a complete transition system specification in the relaxed panth format (see e.g. [12]). The axioms given in Tables 7 and 8 are sound with respect to bisimulation equivalence.

4 Thread-Service Composition

In this section, we extend the thread algebra introduced in Section 3 with thread-service composition, which allows for certain basic actions performed by a thread to be processed by a certain service. This is needed if certain basic actions are

Table 9 Transition rules for cyclic interleaving and deadlock at termination

$$\frac{x_1 \downarrow, \ldots, x_k \downarrow, \langle x_{k+1}, \rho\rangle \xrightarrow{a} \langle x'_{k+1}, \rho'\rangle}{\langle \|(\langle x_1\rangle \curvearrowright \ldots \curvearrowright \langle x_{k+1}\rangle \curvearrowright \alpha), \rho\rangle \xrightarrow{a} \langle \|(\alpha \curvearrowright \langle x'_{k+1}\rangle), \rho'\rangle} \quad (k \geq 0)$$

$$\frac{x_1 \updownarrow, \ldots, x_k \updownarrow, x_l \uparrow, \langle x_{k+1}, \rho\rangle \xrightarrow{a} \langle x'_{k+1}, \rho'\rangle}{\langle \|(\langle x_1\rangle \curvearrowright \ldots \curvearrowright \langle x_{k+1}\rangle \curvearrowright \alpha), \rho\rangle \xrightarrow{a} \langle \|(\alpha \curvearrowright \langle \mathsf{D}\rangle \curvearrowright \langle x'_{k+1}\rangle), \rho'\rangle} \quad (k \geq l > 0)$$

$$\frac{x_1 \downarrow, \ldots, x_k \downarrow}{\|(\langle x_1\rangle \curvearrowright \ldots \curvearrowright \langle x_k\rangle) \downarrow} \qquad \frac{x_1 \updownarrow, \ldots, x_k \updownarrow, x_l \uparrow}{\|(\langle x_1\rangle \curvearrowright \ldots \curvearrowright \langle x_k\rangle) \uparrow} \quad (k \geq l > 0)$$

$$\frac{\langle x, \rho\rangle \xrightarrow{a} \langle x', \rho'\rangle}{\langle \mathsf{S_D}(x), \rho\rangle \xrightarrow{a} \langle \mathsf{S_D}(x'), \rho'\rangle} \qquad \frac{x \updownarrow}{\mathsf{S_D}(x) \uparrow}$$

performed by the thread only for the sake of getting reply values returned by a certain service and that way having itself affected by that service.

For each $f \in \mathcal{F}$, we introduce a *thread-service composition* operator $\_ \mathbin{/_f} \_$. These operators have a thread as first argument and a service as second argument. $P \mathbin{/_f} H$ is the thread that results from processing all basic actions performed by thread P that are of the form $f.m$ by service H. When a basic action $f.m$ performed by thread P is processed by H, it is turned into the action `tau` and postconditional composition is removed in favour of action prefixing on the basis of the reply value produced by H.

A service is represented by a function $H : \mathcal{M}^+ \to \{\mathsf{T}, \mathsf{F}, \mathsf{B}, \mathsf{R}\}$ with the property that $H(\alpha) = \mathsf{B} \Rightarrow H(\alpha \curvearrowright \langle m\rangle) = \mathsf{B}$ and $H(\alpha) = \mathsf{R} \Rightarrow H(\alpha \curvearrowright \langle m\rangle) = \mathsf{R}$ for all $\alpha \in \mathcal{M}^+$ and $m \in \mathcal{M}$. This function is called the *reply* function of the service. Given a reply function H and a method m, the derived reply function of H after processing m, written $\frac{\partial}{\partial m}H$, is defined by $\frac{\partial}{\partial m}H(\alpha) = H(\langle m\rangle \curvearrowright \alpha)$.

The connection between a reply function H and the service represented by it can be understood as follows:

– If $H(\langle m\rangle) = \mathsf{T}$, the request to process command m is accepted by the service, the reply is positive and the service proceeds as $\frac{\partial}{\partial m}H$.
– If $H(\langle m\rangle) = \mathsf{F}$, the request to process command m is accepted by the service, the reply is negative and the service proceeds as $\frac{\partial}{\partial m}H$.
– If $H(\langle m\rangle) = \mathsf{B}$, the request to process command m is not refused by the service, but the processing of m is temporarily blocked. The request will have to wait until the processing of m is not blocked any longer.
– If $H(\langle m\rangle) = \mathsf{R}$, the request to process command m is refused by the service.

The axioms for thread-service composition are given in Table 10. In this table and all subsequent tables with axioms or transition rules in which g occurs, like f, g stands for an arbitrary focus from $\mathcal{F}$. Axiom TSC3 expresses that the action `tau` is always accepted. Axioms TSC5 and TSC6 make it clear that `tau` arises as the residue of processing commands. Therefore, `tau` is not connected to a particular focus, and is always accepted.

Henceforth, we write $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ for $\mathrm{TA}_{\mathrm{fm}}$ extended with the thread-service composition operators and the axioms from Table 10.

Table 10 Axioms for thread-service composition

$\mathsf{S} /_f H = \mathsf{S}$		TSC1
$\mathsf{D} /_f H = \mathsf{D}$		TSC2
$(\mathsf{tau} \circ x) /_f H = \mathsf{tau} \circ (x /_f H)$		TSC3
$(x \unlhd g.m \unrhd y) /_f H = (x /_f H) \unlhd g.m \unrhd (y /_f H)$	if $f \neq g$	TSC4
$(x \unlhd f.m \unrhd y) /_f H = \mathsf{tau} \circ (x /_f \frac{\partial}{\partial m} H)$	if $H(\langle m \rangle) = \mathsf{T}$	TSC5
$(x \unlhd f.m \unrhd y) /_f H = \mathsf{tau} \circ (y /_f \frac{\partial}{\partial m} H)$	if $H(\langle m \rangle) = \mathsf{F}$	TSC6
$(x \unlhd f.m \unrhd y) /_f H = \mathsf{D}$	if $H(\langle m \rangle) \in \{\mathsf{B}, \mathsf{R}\}$	TSC7

Example 2 Let $m, m', m'' \in \mathcal{M}$, and let H be a service such that $H(\alpha \frown \langle m \rangle) = \mathsf{T}$ if $\#_{m'}(\alpha) - \#_{m''}(\alpha) > 0$, $H(\alpha \frown \langle m \rangle) = \mathsf{F}$ if $\#_{m'}(\alpha) - \#_{m''}(\alpha) \leq 0$, $H(\alpha \frown \langle m' \rangle) = \mathsf{T}$ and $H(\alpha \frown \langle m'' \rangle) = \mathsf{T}$, for all $\alpha \in \mathcal{M}^*$. Here $\#_{m'}(\alpha)$ and $\#_{m''}(\alpha)$ denote the number of occurrences of m' and m'', respectively, in α. Then the following equation is easily derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$:

$$(f.m' \circ ((f'.m' \circ \mathsf{S}) \unlhd f.m \unrhd (f''.m'' \circ \mathsf{S}))) /_f H = \mathsf{tau} \circ \mathsf{tau} \circ f'.m' \circ \mathsf{S} \;.$$

This equation shows clearly how the thread $f.m' \circ ((f'.m' \circ \mathsf{S}) \unlhd f.m \unrhd (f''.m'' \circ \mathsf{S}))$ is affected by service H: the processing of $f.m'$ and $f.m$ by H turns these basic actions into tau, and the reply value returned by H after completion of the processing of $f.m$ makes the thread proceed with performing $f'.m'$.

We can prove that each closed term of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ can be reduced to a closed term of BPPA(FM).

Theorem 3 (Elimination) *For all closed terms p of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$, there exists a closed term q of* BPPA(FM) *such that $p = q$ is derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$.*

Proof The proof follows the same lines as the proof of Theorem 2. Here, we have to consider one additional case, viz. $p \equiv p' /_f H$. By the induction hypothesis, there exists a closed term q' of BPPA(FM) such that $p' = q'$. So we are done if we have proved the following lemma:

Let q' be a closed term of BPPA(FM). Then there exists a closed term r' of BPPA(FM) such that $q' /_f H = r'$ is derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$.

We prove this lemma by induction on the depth of q' and case distinction on the structure of q':

- $q' \equiv \mathsf{S}$: $\mathsf{S} /_f H = \mathsf{S}$ by TSC1 and S is a closed term of BPPA(FM).
- $q' \equiv \mathsf{D}$: $\mathsf{D} /_f H = \mathsf{D}$ by TSC2 and D is a closed term of BPPA(FM).
- $q' \equiv \mathsf{tau} \circ q''$: $(\mathsf{tau} \circ q'') /_f H = \mathsf{tau} \circ (q'' /_f H)$ by TSC3. Let r'' be a closed term of BPPA(FM) such that $q'' /_f H = r''$. Such a term exists by the induction hypothesis. Then $\mathsf{tau} \circ r''$ is a closed term of BPPA(FM) and $(\mathsf{tau} \circ q'') /_f H = \mathsf{tau} \circ r''$.
- $q' \equiv q'' \unlhd g.m \unrhd q'''$: We distinguish four cases:

  - $f \neq g$: $(q'' \trianglelefteq g.m \trianglerighteq q''') /_f H = (q'' /_f H) \trianglelefteq g.m \trianglerighteq (q''' /_f H)$ by TSC4. Let r'' and r''' be closed terms of BPPA(FM) such that $q'' /_f H = r''$ and $q''' /_f H = r'''$. Such terms exist by the induction hypothesis. Then $r'' \trianglelefteq g.m \trianglerighteq r'''$ is a closed term of BPPA(FM) and $(q'' \trianglelefteq g.m \trianglerighteq q''') /_f H = r'' \trianglelefteq g.m \trianglerighteq r'''$.
  - $f = g$, $H(\langle m \rangle) = \mathsf{T}$: $(q'' \trianglelefteq g.m \trianglerighteq q''') /_f H = \mathsf{tau} \circ (q'' /_f \frac{\partial}{\partial m} H)$ by TSC5. Let r'' be a closed term of BPPA(FM) such that $q'' /_f \frac{\partial}{\partial m} H = r''$. Such a term exists by the induction hypothesis. Then $\mathsf{tau} \circ r''$ is a closed term of BPPA(FM) and $(q'' \trianglelefteq g.m \trianglerighteq q''') /_f H = \mathsf{tau} \circ r''$.
  - $f = g$, $H(\langle m \rangle) = \mathsf{F}$: This case goes analogous to the previous case.
  - $f = g$, $H(\langle m \rangle) \in \{\mathsf{B}, \mathsf{R}\}$: $(q'' \trianglelefteq g.m \trianglerighteq q''') /_f H = \mathsf{D}$ by TSC7 and D is a closed term of BPPA(FM).

□

The following are useful properties of the deadlock at termination operator in the presence of both cyclic interleaving and thread-service composition which are proved using Theorem 3.

Proposition 5 *For all closed terms $p_1, \ldots, p_n$ of $\mathrm{TA}^{\mathrm{tsc}}_{\mathrm{fm}}$, the following equations are derivable from the axioms of $\mathrm{TA}^{\mathrm{tsc}}_{\mathrm{fm}}$:*

$$\mathsf{S_D}(\mathsf{S_D}(p_1)) = \mathsf{S_D}(p_1) \ , \tag{1}$$

$$\mathsf{S_D}(\|(\langle p_1 \rangle \curvearrowright \ldots \curvearrowright \langle p_n \rangle)) = \|(\langle \mathsf{S_D}(p_1) \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S_D}(p_n) \rangle) \ , \tag{2}$$

$$\mathsf{S_D}(p_1 /_f H) = \mathsf{S_D}(p_1) /_f H \ . \tag{3}$$

Proof By Theorem 3, it is sufficient to prove that these equations are derivable for all closed terms $p_1, \ldots, p_n$ of BPPA(FM). For equations (1) and (2), this is already done in the proof of Proposition 2. For equation (3), we do it by induction on the depth of p_1 and case distinction on the structure of p_1:

- $p_1 \equiv \mathsf{S}$: $\mathsf{S_D}(\mathsf{S} /_f H) = \mathsf{D}$ by TSC1 and S2D1, and $\mathsf{D} = \mathsf{S_D}(\mathsf{S}) /_f H$ by TSC2 and S2D1.
- $p_1 \equiv \mathsf{D}$: $\mathsf{S_D}(\mathsf{D} /_f H) = \mathsf{D}$ by TSC2 and S2D2, and $\mathsf{D} = \mathsf{S_D}(\mathsf{D}) /_f H$ by TSC2 and S2D2.
- $p_1 \equiv \mathsf{tau} \circ p_1'$: $\mathsf{S_D}((\mathsf{tau} \circ p_1') /_f H) = \mathsf{tau} \circ \mathsf{S_D}(p_1' /_f H)$ by TSC3 and S2D3, $\mathsf{tau} \circ \mathsf{S_D}(p_1' /_f H) = \mathsf{tau} \circ (\mathsf{S_D}(p_1') /_f H)$ by the induction hypothesis, and $\mathsf{tau} \circ (\mathsf{S_D}(p_1') /_f H) = \mathsf{S_D}(\mathsf{tau} \circ p_1') /_f H$ by TSC3 and S2D3.
- $p_1 \equiv p_1' \trianglelefteq g.m \trianglerighteq p_1''$: We distinguish four cases:
  - $f \neq g$: $\mathsf{S_D}((p_1' \trianglelefteq g.m \trianglerighteq p_1'') /_f H) = \mathsf{S_D}(p_1' /_f H) \trianglelefteq g.m \trianglerighteq \mathsf{S_D}(p_1'' /_f H)$ by TSC4 and S2D4, $\mathsf{S_D}(p_1' /_f H) \trianglelefteq g.m \trianglerighteq \mathsf{S_D}(p_1'' /_f H) = (\mathsf{S_D}(p_1') /_f H) \trianglelefteq g.m \trianglerighteq (\mathsf{S_D}(p_1'') /_f H)$ by the induction hypothesis, and $(\mathsf{S_D}(p_1') /_f H) \trianglelefteq g.m \trianglerighteq (\mathsf{S_D}(p_1'') /_f H) = \mathsf{S_D}(p_1' \trianglelefteq g.m \trianglerighteq p_1'') /_f H$ by TSC4 and S2D4.
  - $f = g$, $H(\langle m \rangle) = \mathsf{T}$: $\mathsf{S_D}((p_1' \trianglelefteq g.m \trianglerighteq p_1'') /_f H) = \mathsf{tau} \circ \mathsf{S_D}(p_1' /_f \frac{\partial}{\partial m} H)$ by TSC5 and S2D3, $\mathsf{tau} \circ \mathsf{S_D}(p_1' /_f \frac{\partial}{\partial m} H) = \mathsf{tau} \circ (\mathsf{S_D}(p_1') /_f \frac{\partial}{\partial m} H)$ by the induction hypothesis, and $\mathsf{tau} \circ (\mathsf{S_D}(p_1') /_f \frac{\partial}{\partial m} H) = \mathsf{S_D}(p_1' \trianglelefteq g.m \trianglerighteq p_1'') /_f H$ by TSC5 and S2D4.
  - $f = g$, $H(\langle m \rangle) = \mathsf{F}$: This case goes analogous to the previous case.

– $f = g$, $H(\langle m \rangle) \in \{\mathsf{B}, \mathsf{R}\}$: $\mathsf{S}_\mathsf{D}((p'_1 \trianglelefteq g.m \trianglerighteq p''_1) /_f H) = \mathsf{D}$ by TSC7 and S2D2, and $\mathsf{D} = \mathsf{S}_\mathsf{D}(p'_1 \trianglelefteq g.m \trianglerighteq p''_1) /_f H$ by TSC7 and S2D4.

□

We extend $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ with guarded recursion as in the case of $\mathrm{TA}_{\mathrm{fm}}$. Systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ and guardedness of those are defined as in the case of $\mathrm{TA}_{\mathrm{fm}}$, but with $\mathrm{TA}_{\mathrm{fm}}$ everywhere replaced by $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$.

Henceforth, we will write $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$+REC for $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ extended with the constants for solutions of guarded systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ and the axioms RDP and RSP from Table 2.

Theorem 3 states that the strategic interleaving operator for cyclic interleaving, the deadlock at termination operator and the thread-service composition operators can be eliminated from closed terms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$. It does not state anything about closed terms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$+REC. Propositions 3 and 4, concerning the case where the operand of the strategic interleaving operator for cyclic interleaving is a sequence of constants for solutions of guarded systems of recursion equations over BPPA(FM) and the case where the operand of the deadlock at termination operator is such a constant, go through in the presence of the thread-service composition operators. The following proposition concerns the case where the first operand of a thread-service composition operator is such a constant.

Proposition 6 *Let E' be a guarded system of recursion equations over* BPPA(FM)*, and let $X \in \mathrm{V}(E')$. Moreover, let f be a focus and let H be a reply function. Then there exists a guarded system of recursion equations E over* BPPA(FM) *and a variable $Y \in \mathrm{V}(E)$ such that $\langle X|E' \rangle /_f H = \langle Y|E \rangle$ is derivable from the axioms of* $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$+REC.

Proof Without loss of generality, we may assume that all occurrences of variables in the right-hand sides of the equations in E' are guarded (see Remark 1). Let $\mathcal{H}$ be the set inductively defined by the following rules: (i) $H \in \mathcal{H}$; (ii) if $m \in \mathcal{M}$ and $H' \in \mathcal{H}$, then $\frac{\partial}{\partial m} H' \in \mathcal{H}$. We take an injective function Y that maps each pair in $\mathrm{V}(E') \times \mathcal{H}$ to a variable not in $\mathrm{V}(E')$, and define, guided by axioms TSC1–TSC7, the following guarded system of recursion equations:

$$
\begin{aligned}
E = {} & \{Y(X',H') = \mathsf{S} \mid X' = \mathsf{S} \in E' \wedge H' \in \mathcal{H}\} \\
& \cup \{Y(X',H') = \mathsf{D} \mid X' = \mathsf{D} \in E' \wedge H' \in \mathcal{H}\} \\
& \cup \{Y(X',H') = \mathsf{tau} \circ Y(X'',H') \mid X' = \mathsf{tau} \circ X'' \in E' \wedge H' \in \mathcal{H}\} \\
& \cup \{Y(X',H') = Y(X'',H') \trianglelefteq g.m \trianglerighteq Y(X''',H') \mid \\
& \qquad X' = X'' \trianglelefteq g.m \trianglerighteq X''' \in E' \wedge f \neq g \wedge H' \in \mathcal{H}\} \\
& \cup \{Y(X',H') = \mathsf{tau} \circ Y(X'', \tfrac{\partial}{\partial m} H') \mid \\
& \qquad \exists X''' \bullet (X' = X'' \trianglelefteq f.m \trianglerighteq X''' \in E' \wedge H'(\langle m \rangle) = \mathsf{T} \wedge H' \in \mathcal{H})\} \\
& \cup \{Y(X',H') = \mathsf{tau} \circ Y(X''', \tfrac{\partial}{\partial m} H') \mid \\
& \qquad \exists X'' \bullet (X' = X'' \trianglelefteq f.m \trianglerighteq X''' \in E' \wedge H'(\langle m \rangle) = \mathsf{F} \wedge H' \in \mathcal{H})\} \\
& \cup \{Y(X',H') = \mathsf{D} \mid \exists m, X'', X''' \bullet \\
& \qquad\quad (X' = X'' \trianglelefteq f.m \trianglerighteq X''' \in E' \wedge H'(\langle m \rangle) \in \{\mathsf{B}, \mathsf{R}\} \wedge H' \in \mathcal{H})\} .
\end{aligned}
$$

Table 11 Transition rules for thread-service composition

$$\frac{\langle x, \rho\rangle \xrightarrow{\mathsf{tau}} \langle x', \rho'\rangle}{\langle x \mathbin{/_f} H, \rho\rangle \xrightarrow{\mathsf{tau}} \langle x' \mathbin{/_f} H, \rho'\rangle} \qquad \frac{\langle x, \rho\rangle \xrightarrow{g.m} \langle x', \rho'\rangle}{\langle x \mathbin{/_f} H, \rho\rangle \xrightarrow{g.m} \langle x' \mathbin{/_f} H, \rho'\rangle}\; f \neq g$$

$$\frac{\langle x, \rho\rangle \xrightarrow{f.m} \langle x', \rho'\rangle}{\langle x \mathbin{/_f} H, \rho\rangle \xrightarrow{\mathsf{tau}} \langle x' \mathbin{/_f} \frac{\partial}{\partial m} H, \rho'\rangle}\; H(\langle m\rangle) \in \{\mathsf{T}, \mathsf{F}\},\, (f.m, H(\langle m\rangle)) \in \rho(\langle\,\rangle)$$

$$\frac{\langle x, \rho\rangle \xrightarrow{f.m} \langle x', \rho'\rangle}{x \mathbin{/_f} H\!\uparrow}\; H(\langle m\rangle) \in \{\mathsf{B}, \mathsf{R}\} \qquad \frac{x\!\downarrow}{x \mathbin{/_f} H\!\downarrow} \qquad \frac{x\!\uparrow}{x \mathbin{/_f} H\!\uparrow}$$

If we replace in E, for all $X' \in \mathrm{V}(E')$ and all $H' \in \mathcal{H}$, all occurrences of $Y(X', H')$ by $\langle X'|E'\rangle \mathbin{/_f} H$, then each of the resulting equations is derivable by first applying RDP and then applying one of TSC1–TSC7. Hence, $\langle X|E'\rangle \mathbin{/_f} H$ is a solution of E for $Y(X, H)$. From this, it follows by RSP that $\langle X|E'\rangle \mathbin{/_f} H = \langle Y(X, H)|E\rangle$. □

The structural operational semantics of $\mathrm{TA}^{\mathrm{tsc}}_{\mathrm{fm}}$ is described by the transition rules given in Tables 4, 9 and 11.

Bisimulation equivalence is also a congruence with respect to the thread-service composition operators. This follows immediately from the fact that the transition rules for these operators are in the path format. The axioms given in Table 10 are sound with respect to bisimulation equivalence.

5 Guarding Tests

In this section, we extend the thread algebra developed in Sections 3 and 4 with guarding tests. Guarding tests are basic actions meant to verify whether a service will accept the request to process a certain method now, and if not so whether it will be accepted after some time. Guarding tests allow for dealing with delayed processing and exception handling as will be shown in Section 6.

We extend the set of basic actions. For the set of basic actions $\mathcal{A}$, we now take the set $\mathit{FM}^{\mathrm{gt}} = \{f.m, f?m, f??m \mid f \in \mathcal{F}, m \in \mathcal{M}\}$. Basic actions of the forms $f?m$ and $f??m$ will be called *guarding tests*. Performing a basic action $f?m$ is taken as making the request to the service named f to reply whether it will accept the request to process method m now. The reply is positive if the service will accept that request now, and otherwise it is negative. Performing a basic action $f??m$ is taken as making the request to the service named f to reply whether it will accept the request to process method m now or after some time. The reply is positive if the service will accept that request now or after some time, and otherwise it is negative.

A service may be local to a single thread, local to a multi-thread, local to a host, or local to a network. A service local to a multi-thread is shared by all threads from which the multi-thread is composed, etc. Henceforth, to simplify matters, it is assumed that each thread, each multi-thread, each host, and each network has a

Table 12 Additional axioms for cyclic interleaving & deadlock at termination

$\Vert(\langle x \unlhd f?m \unrhd y\rangle \curvearrowright \alpha) = \Vert(\langle x\rangle \curvearrowright \alpha) \unlhd f?m \unrhd \Vert(\alpha \curvearrowright \langle y\rangle)$	CSI6
$\Vert(\langle x \unlhd f??m \unrhd y\rangle \curvearrowright \alpha) = \Vert(\langle x\rangle \curvearrowright \alpha) \unlhd f??m \unrhd \Vert(\alpha \curvearrowright \langle y\rangle)$	CSI7
$\mathsf{S_D}(x \unlhd f?m \unrhd y) = \mathsf{S_D}(x) \unlhd f?m \unrhd \mathsf{S_D}(y)$	S2D5
$\mathsf{S_D}(x \unlhd f??m \unrhd y) = \mathsf{S_D}(x) \unlhd f??m \unrhd \mathsf{S_D}(y)$	S2D6

Table 13 Additional axioms for thread-service composition

$(x \unlhd g?m \unrhd y) /_f H = (x /_f H) \unlhd g?m \unrhd (y /_f H)$	if $f \neq g$	TSC8
$(x \unlhd f?m \unrhd y) /_f H = \mathsf{tau} \circ (x /_f H)$	if $H(\langle m\rangle) \in \{\mathsf{T}, \mathsf{F}\}$	TSC9
$(x \unlhd f?m \unrhd y) /_f H = \mathsf{tau} \circ (y /_f H)$	if $H(\langle m\rangle) = \mathsf{B} \land f \neq \mathsf{t}$	TSC10
$(x \unlhd f?m \unrhd y) /_f H = \mathsf{D}$	if $(H(\langle m\rangle) = \mathsf{B} \land f = \mathsf{t}) \lor$ $H(\langle m\rangle) = \mathsf{R}$	TSC11
$(x \unlhd g??m \unrhd y) /_f H = (x /_f H) \unlhd g??m \unrhd (y /_f H)$	if $f \neq g$	TSC12
$(x \unlhd f??m \unrhd y) /_f H = \mathsf{tau} \circ (x /_f H)$	if $H(\langle m\rangle) \in \{\mathsf{T}, \mathsf{F}, \mathsf{B}\}$	TSC13
$(x \unlhd f??m \unrhd y) /_f H = \mathsf{tau} \circ (y /_f H)$	if $H(\langle m\rangle) = \mathsf{R}$	TSC14

unique local service. Moreover, it is assumed that $\mathsf{t}, \mathsf{p}, \mathsf{h}, \mathsf{n} \in \mathcal{F}$. Below, the foci t, p, h and n play a special role:

- for each thread, t is the focus of its unique local service;
- for each multi-thread, p is the focus of its unique local service;
- for each host, h is the focus of its unique local service;
- for each network, n is the focus of its unique local service.

As explained below, it happens that not only thread-service composition but also cyclic interleaving has to be adapted to the presence of guarding tests.

The additional axioms for cyclic interleaving and deadlock at termination in the presence of guarding tests are given in Table 12. Axioms CSI6 and CSI7 state that:

- after a positive reply on $f?m$ or $f??m$, the same thread proceeds with its next basic action; and thus it is prevented that meanwhile other threads can cause a state change to a state in which the processing of m is blocked (and $f?m$ would not reply positively) or the processing of m is refused (and both $f?m$ and $f??m$ would not reply positively);
- after a negative reply on $f?m$ or $f??m$, the same thread does not proceed with it; and thus it is prevented that other threads cannot make progress.

Without this difference, Theorem 5 in Section 7 would not go through.

The additional axioms for thread-service composition in the presence of guarding tests are given in Table 13. Axioms TSC10 and TSC11 are crucial. The point is that, if the local service of a thread is in a state in which the processing of method m is blocked, no other thread can raise that state. Consequently, if the processing of m is blocked, it is blocked forever.

Henceforth, we write $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ for $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc}}$ extended with a postconditional composition operator for each guarding test and the axioms from Tables 12 and 13.

We can prove that each closed term of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ can be reduced to a closed term of BPPA(FM^{gt}).

Theorem 4 (Elimination) *For all closed terms p of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$, there exists a closed term q of BPPA(FM^{gt}) such that $p = q$ is derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$.*

Proof The proof follows the same lines as the proof of Theorem 3. Here, we have to consider two additional cases, viz. $p \equiv p' \trianglelefteq f?m \trianglerighteq p''$ and $p \equiv p' \trianglelefteq f??m \trianglerighteq p''$. These cases go the same as the case $p \equiv p' \trianglelefteq f.m \trianglerighteq p''$. In the lemma for the case $p \equiv \mathsf{S}_{\mathsf{D}}(p')$, we have to consider the additional cases $q' \equiv q'' \trianglelefteq f?m \trianglerighteq q'''$ and $q' \equiv q'' \trianglelefteq f??m \trianglerighteq q'''$. These cases go the same as the case $q' \equiv q'' \trianglelefteq f.m \trianglerighteq q'''$. In the lemma for the case $p \equiv \|(\alpha)$, we have to consider the additional cases $q'_1 \equiv q''_1 \trianglelefteq f?m \trianglerighteq q'''_1$ and $q'_1 \equiv q''_1 \trianglelefteq f??m \trianglerighteq q'''_1$. These cases go analogous to the case $q'_1 \equiv q''_1 \trianglelefteq f.m \trianglerighteq q'''_1$. In the lemma for the case $p \equiv p' /_f H$, we have to consider the additional cases $q' \equiv q'' \trianglelefteq g?m \trianglerighteq q'''$ and $q' \equiv q'' \trianglelefteq g??m \trianglerighteq q'''$. These cases go similar to the case $q' \equiv q'' \trianglelefteq g.m \trianglerighteq q'''$. $\square$

In other words, Theorem 3 goes through in the presence of guarding tests.

We extend $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ with guarded recursion as in the case of $\mathrm{TA}_{\mathrm{fm}}$. Systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ and guardedness of those are defined as in the case of $\mathrm{TA}_{\mathrm{fm}}$, but with $\mathrm{TA}_{\mathrm{fm}}$ everywhere replaced by $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$.

Henceforth, we will write $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$+REC for $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ extended with the constants for solutions of guarded systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ and the axioms RDP and RSP from Table 2.

Example 3 Let $f \in \mathcal{F}$ be such that $f \neq \mathsf{t}$, let $m, m', m'' \in \mathcal{M}$, and let H be a service such that $H(\alpha \curvearrowright \langle m \rangle) = \mathsf{T}$ if $\#_{m'}(\alpha) - \#_{m''}(\alpha) > 0$, $H(\alpha \curvearrowright \langle m \rangle) = \mathsf{B}$ if $\#_{m'}(\alpha) - \#_{m''}(\alpha) \leq 0$, $H(\alpha \curvearrowright \langle m' \rangle) = \mathsf{T}$ and $H(\alpha \curvearrowright \langle m'' \rangle) = \mathsf{T}$, for all $\alpha \in \mathcal{M}^*$. Moreover, let E be the guarded system of recursion equations that consists of the equation $X = ((f'.m' \circ \mathsf{S}) \trianglelefteq f.m \trianglerighteq (f''.m'' \circ \mathsf{S})) \trianglelefteq f?m \trianglerighteq X$. Then the following equations are easily derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$+REC:

$$\|(\langle (f'.m' \circ \mathsf{S}) \trianglelefteq f.m \trianglerighteq (f''.m'' \circ \mathsf{S}) \rangle \curvearrowright \langle f.m' \circ \mathsf{S} \rangle) /_f H = \mathsf{tau} \circ \mathsf{D} \;,$$

$$\|(\langle \langle X|E \rangle \rangle \curvearrowright \langle f.m' \circ \mathsf{S} \rangle) /_f H = \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{tau} \circ f'.m' \circ \mathsf{S} \;.$$

The first basic action performed by $\|(\langle (f'.m' \circ \mathsf{S}) \trianglelefteq f.m \trianglerighteq (f''.m'' \circ \mathsf{S}) \rangle \curvearrowright \langle f.m' \circ \mathsf{S} \rangle)$ is $f.m$. Because $H(\langle m \rangle) = \mathsf{B}$, the processing of $f.m$ by H leads to D. The first basic action performed by $\|(\langle \langle X|E \rangle \rangle \curvearrowright \langle f.m' \circ \mathsf{S} \rangle)$ is $f?m$. Because $H(\langle m \rangle) = \mathsf{B}$ and $f \neq \mathsf{t}$, next $f.m'$ is performed and thereafter $f?m$ is performed again. Because $\frac{\partial}{\partial m'} H(\langle m \rangle) = \mathsf{T}$, next $f.m$ is performed and thereafter $f'.m'$ is performed.

Just like Theorem 3, Propositions 3, 4 and 6 go through in the presence of guarding tests. The proofs follow the same lines as before, but, like in the proof of Theorem 4, we have to take into account that two additional kinds of basic actions may occur in guarded systems of recursion equations.

The additional transition rules for cyclic interleaving and deadlock at termination in the presence of guarding tests are given in Table 14, where γ stands for an

Table 14 Additional transition rules for cyclic interleaving & deadlock at termination

$$\frac{x_1 {\downarrow}, \ldots, x_k {\downarrow}, \langle x_{k+1}, \rho \rangle \xrightarrow{\gamma} \langle x'_{k+1}, \rho' \rangle}{\langle \|(\langle x_1 \rangle \frown \ldots \frown \langle x_{k+1} \rangle \frown \alpha), \rho \rangle \xrightarrow{\gamma} \langle \|(\langle x'_{k+1} \rangle \frown \alpha), \rho' \rangle} \; (\alpha, \mathsf{T}) \in \rho(\langle\,\rangle) \qquad (k \geq 0)$$

$$\frac{x_1 {\updownarrow}, \ldots, x_k {\updownarrow}, x_l {\uparrow}, \langle x_{k+1}, \rho \rangle \xrightarrow{\gamma} \langle x'_{k+1}, \rho' \rangle}{\langle \|(\langle x_1 \rangle \frown \ldots \frown \langle x_{k+1} \rangle \frown \alpha), \rho \rangle \xrightarrow{\gamma} \langle \|(\langle x'_{k+1} \rangle \frown \alpha \frown \langle \mathsf{D} \rangle), \rho' \rangle} \; (\alpha, \mathsf{T}) \in \rho(\langle\,\rangle) \quad (k \geq l > 0)$$

$$\frac{x_1 {\downarrow}, \ldots, x_k {\downarrow}, \langle x_{k+1}, \rho \rangle \xrightarrow{\gamma} \langle x'_{k+1}, \rho' \rangle}{\langle \|(\langle x_1 \rangle \frown \ldots \frown \langle x_{k+1} \rangle \frown \alpha), \rho \rangle \xrightarrow{\gamma} \langle \|(\alpha \frown \langle x'_{k+1} \rangle), \rho' \rangle} \; (\alpha, \mathsf{F}) \in \rho(\langle\,\rangle) \qquad (k \geq 0)$$

$$\frac{x_1 {\updownarrow}, \ldots, x_k {\updownarrow}, x_l {\uparrow}, \langle x_{k+1}, \rho \rangle \xrightarrow{\gamma} \langle x'_{k+1}, \rho' \rangle}{\langle \|(\langle x_1 \rangle \frown \ldots \frown \langle x_{k+1} \rangle \frown \alpha), \rho \rangle \xrightarrow{\gamma} \langle \|(\alpha \frown \langle \mathsf{D} \rangle \frown \langle x'_{k+1} \rangle), \rho' \rangle} \; (\alpha, \mathsf{F}) \in \rho(\langle\,\rangle) \quad (k \geq l > 0)$$

$$\frac{\langle x, \rho \rangle \xrightarrow{\gamma} \langle x', \rho' \rangle}{\langle \mathsf{S_D}(x), \rho \rangle \xrightarrow{\gamma} \langle \mathsf{S_D}(x'), \rho' \rangle}$$

Table 15 Additional transition rules for thread-service composition

$$\frac{\langle x, \rho \rangle \xrightarrow{g?m} \langle x', \rho' \rangle}{\langle x /_f H, \rho \rangle \xrightarrow{g?m} \langle x' /_f H, \rho' \rangle} \; f \neq g$$

$$\frac{\langle x, \rho \rangle \xrightarrow{f?m} \langle x', \rho' \rangle}{\langle x /_f H, \rho \rangle \xrightarrow{\mathsf{tau}} \langle x' /_f H, \rho' \rangle} \; H(\langle m \rangle) \in \{\mathsf{T}, \mathsf{F}\},\ (f?m, \mathsf{T}) \in \rho(\langle\,\rangle)$$

$$\frac{\langle x, \rho \rangle \xrightarrow{f?m} \langle x', \rho' \rangle}{\langle x /_f H, \rho \rangle \xrightarrow{\mathsf{tau}} \langle x' /_f H, \rho' \rangle} \; H(\langle m \rangle) = \mathsf{B},\ f \neq \mathsf{t},\ (f?m, \mathsf{F}) \in \rho(\langle\,\rangle)$$

$$\frac{\langle x, \rho \rangle \xrightarrow{\mathsf{t}?m} \langle x', \rho' \rangle}{x /_{\mathsf{t}} H {\uparrow}} \; H(\langle m \rangle) = \mathsf{B} \qquad \frac{\langle x, \rho \rangle \xrightarrow{f?m} \langle x', \rho' \rangle}{x /_f H {\uparrow}} \; H(\langle m \rangle) = \mathsf{R}$$

$$\frac{\langle x, \rho \rangle \xrightarrow{g??m} \langle x', \rho' \rangle}{\langle x /_f H, \rho \rangle \xrightarrow{g??m} \langle x' /_f H, \rho' \rangle} \; f \neq g$$

$$\frac{\langle x, \rho \rangle \xrightarrow{f??m} \langle x', \rho' \rangle}{\langle x /_f H, \rho \rangle \xrightarrow{\mathsf{tau}} \langle x' /_f H, \rho' \rangle} \; H(\langle m \rangle) \in \{\mathsf{T}, \mathsf{F}, \mathsf{B}\},\ (f??m, \mathsf{T}) \in \rho(\langle\,\rangle)$$

$$\frac{\langle x, \rho \rangle \xrightarrow{f??m} \langle x', \rho' \rangle}{\langle x /_f H, \rho \rangle \xrightarrow{\mathsf{tau}} \langle x' /_f H, \rho' \rangle} \; H(\langle m \rangle) = \mathsf{R},\ (f??m, \mathsf{F}) \in \rho(\langle\,\rangle)$$

arbitrary basic action from the set $\{f?m, f??m \mid f \in \mathcal{F}, m \in \mathcal{M}\}$. The additional transition rules for thread-service composition in the presence of guarding tests are given in Table 15.

Bisimulation equivalence remains a congruence with respect to these operators. The axioms given in Tables 12 and 13 are sound with respect to bisimulation equivalence.

Table 16 Axioms for delayed processing and exception handling

$x \unlhd f!m \unrhd y = (x \unlhd f.m \unrhd y) \unlhd f?m \unrhd (x \unlhd f!m \unrhd y)$	DP
$x \unlhd f.m\,[y] \unrhd z = (x \unlhd f.m \unrhd z) \unlhd f??m \unrhd y$	EH1
$x \unlhd f!m\,[y] \unrhd z = ((x \unlhd f.m \unrhd z) \unlhd f?m \unrhd (x \unlhd f!m\,[y] \unrhd z)) \unlhd f??m \unrhd y$	EH2

6 Delayed Processing and Exception Handling

We go on to show how guarding tests can be used to express postconditional composition with delayed processing and postconditional composition with exception handling.

For postconditional composition with delayed processing, we extend the set of basic actions with the set $\{f!m \mid f \in \mathcal{F}, m \in \mathcal{M}\}$. Performing a basic action $f!m$ is like performing $f.m$, but in case processing of the command m is temporarily blocked, it is automatically delayed until the blockade is over.

For postconditional composition with exception handling, we introduce the notations $x \unlhd f.m\,[y] \unrhd z$ and $x \unlhd f!m\,[y] \unrhd z$. The intuition for $x \unlhd f.m\,[y] \unrhd z$ is that $x \unlhd f.m \unrhd z$ is tried, but y is done instead in the exceptional case that $x \unlhd f.m \unrhd z$ fails because the request to process m is refused. The intuition for $x \unlhd f!m\,[y] \unrhd z$ is that $x \unlhd f!m \unrhd z$ is tried, but y is done instead in the exceptional case that $x \unlhd f!m \unrhd z$ fails because the request to process m is refused. The processing of m may first be blocked and thereafter be refused; in that case, y is done instead as well.

The defining axioms for postconditional composition with delayed processing and the two forms of postconditional composition with exception handling are given in Table 16. Axiom DP guarantees that $f.m$ is only performed if $f?m$ yields a positive reply. Axioms EH1 and EH2 guarantee that $f.m$ is only performed if $f??m$ yields a positive reply. An alternative to axiom EH2 is

$$x \unlhd f!m\,[y] \unrhd z = ((x \unlhd f.m \unrhd z) \unlhd f?m \unrhd (x \unlhd f!m \unrhd z)) \unlhd f??m \unrhd y\ .$$

In that case, y is only done if the processing of m is refused immediately.

From DP, EH1–EH2 and CSI6–CSI7 (Table 12), it follows immediately that

$$\|(\langle x \unlhd f!m \unrhd y\rangle \frown \alpha) = \|(\langle x \unlhd f.m \unrhd y\rangle \frown \alpha) \unlhd f?m \unrhd \|(\alpha \frown \langle x \unlhd f!m \unrhd y\rangle)\ ,$$

$$\|(\langle x \unlhd f.m\,[y] \unrhd z\rangle \frown \alpha) = \|(\langle x \unlhd f.m \unrhd z\rangle \frown \alpha) \unlhd f??m \unrhd \|(\alpha \frown \langle y\rangle)\ ,$$

$$\begin{aligned}&\|(\langle x \unlhd f!m\,[y] \unrhd z\rangle \frown \alpha)\\ &= (\|(\langle x \unlhd f.m \unrhd z\rangle \frown \alpha) \unlhd f?m \unrhd \|(\alpha \frown \langle x \unlhd f!m\,[y] \unrhd z\rangle)) \unlhd f??m \unrhd \|(\alpha \frown \langle y\rangle)\ .\end{aligned}$$

These equations give a clear picture of the mechanisms for delayed processing and exception handling.

Henceforth, we write $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$ for $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt}}$ extended with the postconditional composition operators for delayed processing and exception handling and the axioms from Table 16.

Table 17 Transition rules for delayed processing and exception handling

$$\frac{\langle (x \unlhd f.m \unrhd y) \unlhd f?m \unrhd (x \unlhd f!m \unrhd y), \rho\rangle \xrightarrow{a} \langle z', \rho'\rangle}{\langle x \unlhd f!m \unrhd y, \rho\rangle \xrightarrow{a} \langle z', \rho'\rangle}$$

$$\frac{\langle (x \unlhd f.m \unrhd z) \unlhd f??m \unrhd y, \rho\rangle \xrightarrow{a} \langle u', \rho'\rangle}{\langle x \unlhd f.m\,[y] \unrhd z, \rho\rangle \xrightarrow{a} \langle u', \rho'\rangle}$$

$$\frac{\langle ((x \unlhd f.m \unrhd z) \unlhd f?m \unrhd (x \unlhd f!m\,[y] \unrhd z)) \unlhd f??m \unrhd y, \rho\rangle \xrightarrow{a} \langle u', \rho'\rangle}{\langle x \unlhd f!m\,[y] \unrhd z, \rho\rangle \xrightarrow{a} \langle u', \rho'\rangle}$$

We extend $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$ with guarded recursion as in the case of $\mathrm{TA}_{\mathrm{fm}}$. Systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$ and guardedness of those are defined as in the case of $\mathrm{TA}_{\mathrm{fm}}$, but with $\mathrm{TA}_{\mathrm{fm}}$ everywhere replaced by $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$.

Henceforth, we will also write $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$+REC for $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$ extended with the constants for solutions of guarded systems of recursion equations over $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$ and the axioms RDP and RSP from Table 2.

Example 4 Let H be as in Example 3. Then the following equations are easily derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$:

$$\begin{aligned}
&\|(\langle (f'.m' \circ \mathsf{S}) \unlhd f.m \unrhd (f''.m'' \circ \mathsf{S})\rangle \curvearrowright \langle f.m' \circ \mathsf{S}\rangle) /_f H \\
&= \mathsf{tau} \circ \mathsf{D}\ ,
\end{aligned}$$

$$\begin{aligned}
&\|(\langle (f'.m' \circ \mathsf{S}) \unlhd f!m \unrhd (f''.m'' \circ \mathsf{S})\rangle \curvearrowright \langle f.m' \circ \mathsf{S}\rangle) /_f H \\
&= \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{tau} \circ f'.m' \circ \mathsf{S}\ .
\end{aligned}$$

The resemblance with the equations from Example 3 is not accidental: the equation $\langle X|E\rangle = (f'.m' \circ \mathsf{S}) \unlhd f!m \unrhd (f''.m'' \circ \mathsf{S})$, in which E is the guarded system of recursion equations from Example 3, is derivable from the axioms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$+REC.

The additional transition rules for postconditional composition with delayed processing and postconditional composition with exception handling are given in Table 17.

Bisimulation equivalence is a congruence with respect to these operators. The axioms given in Table 16 are sound with respect to bisimulation equivalence.

7 A Formal Design Prototype

In this section, we show that the thread algebra developed in Sections 3–6 can be used to develop a simplified, formal representation schema of the design of systems that consist of several multi-threaded programs on various hosts in different networks and to verify a property of all systems designed according to the schema.

We propose to use the term *formal design prototype* for such a schema. The presented schema can be useful in understanding certain aspects of the systems with which it is concerned.

The set of *basic thread expressions*, with typical element P, is defined by

$$\begin{aligned} P \ ::= \ & \mathsf{D} \mid \mathsf{S} \mid P \trianglelefteq f.m \trianglerighteq P \mid P \trianglelefteq f!m \trianglerighteq P \mid \\ & P \trianglelefteq f.m\,[P] \trianglerighteq P \mid P \trianglelefteq f!m\,[P] \trianglerighteq P \mid \langle X|E\rangle \;, \end{aligned}$$

where $f \in \mathcal{F}$, $m \in \mathcal{M}$ and $\langle X|E\rangle$ is a constant standing for the unique solution for variable X of a guarded system of recursion equations E in which the right-hand sides of the equations are basic thread expressions in which variables may occur wherever basic thread expressions are expected. Thus, the use of guarding tests, i.e. basic actions of the forms $f?m$ and $f??m$, is restricted to their intended use.

A thread vector in which each thread has its local service is of the form

$$\langle P_1 \mathbin{/_{\mathsf{t}}} TLS_1\rangle \frown \ldots \frown \langle P_{l_t} \mathbin{/_{\mathsf{t}}} TLS_{l_t}\rangle \;,$$

where $P_1, \ldots, P_{l_t}$ are basic thread expressions, and $TLS_1, \ldots, TLS_{l_t}$ are local services for threads. The local service of a thread does nothing else but maintaining local data for the thread. A multi-thread vector in which each multi-thread has its local service is of the form

$$\langle \|(TV_1) \mathbin{/_{\mathsf{p}}} PLS_1\rangle \frown \ldots \frown \langle \|(TV_{l_p}) \mathbin{/_{\mathsf{p}}} PLS_{l_p}\rangle \;,$$

where $TV_1, \ldots, TV_{l_p}$ are thread vectors in which each thread has its local service, and $PLS_1, \ldots, PLS_{l_p}$ are local services for multi-threads. The local service of a multi-thread maintains shared data of the threads from which the multi-thread is composed. A typical example of such data are Java pipes. A host behaviour vector in which each host has its local service is of the form

$$\langle \|(PV_1) \mathbin{/_{\mathsf{h}}} HLS_1\rangle \frown \ldots \frown \langle \|(PV_{l_h}) \mathbin{/_{\mathsf{h}}} HLS_{l_h}\rangle \;,$$

where $PV_1, \ldots, PV_{l_h}$ are multi-thread vectors in which each multi-thread has its local service, and $HLS_1, \ldots, HLS_{l_h}$ are local services for hosts. The local service of a host maintains shared data of the multi-threads on the host. A typical example of such data are the files connected with Unix sockets used for data transfer between multi-threads on the same host. A network behaviour vector in which each network has its local service is of the form

$$\langle \|(HV_1) \mathbin{/_{\mathsf{n}}} NLS_1\rangle \frown \ldots \frown \langle \|(HV_{l_n}) \mathbin{/_{\mathsf{n}}} NLS_{l_n}\rangle \;,$$

where $HV_1, \ldots, HV_{l_n}$ are host behaviour vectors in which each host has its local service, and $NLS_1, \ldots, NLS_{l_n}$ are local services for networks. The local service of a network maintains shared data of the hosts in the network. A typical example of such data are the files connected with Unix sockets used for data transfer between different hosts in the same network.

The behaviour of a system that consist of several multi-threaded programs on various hosts in different networks is described by an expression of the form

Table 18 Definition of simulation relation

$\mathsf{S}\ \mathsf{sim}\ x$
$\mathsf{D}\ \mathsf{sim}\ x$
$x\ \mathsf{sim}\ y \land x\ \mathsf{sim}\ z \Rightarrow x\ \mathsf{sim}\ y \trianglelefteq a \trianglerighteq z$
$x\ \mathsf{sim}\ y \land z\ \mathsf{sim}\ w \Rightarrow x \trianglelefteq a \trianglerighteq z\ \mathsf{sim}\ y \trianglelefteq a \trianglerighteq w$

$\|(NV)$, where NV is a network behaviour vector in which each network has its local service. A typical example is the case where NV is an expression of the form

$$\begin{aligned}&\|(\langle\|(\langle\|(\langle P_1 \mathbin{/_{\mathsf{t}}} TLS_1\rangle \curvearrowright \langle P_2 \mathbin{/_{\mathsf{t}}} TLS_2\rangle) \mathbin{/_{\mathsf{p}}} PLS_1\rangle \curvearrowright\\ &\qquad \langle\|(\langle P_3 \mathbin{/_{\mathsf{t}}} TLS_3\rangle \curvearrowright \langle P_4 \mathbin{/_{\mathsf{t}}} TLS_4\rangle \curvearrowright \langle P_5 \mathbin{/_{\mathsf{t}}} TLS_5\rangle) \mathbin{/_{\mathsf{p}}} PLS_2\rangle) \mathbin{/_{\mathsf{h}}} HLS_1\rangle \curvearrowright\\ &\quad \langle\|(\langle\|(\langle P_6 \mathbin{/_{\mathsf{t}}} TLS_6\rangle) \mathbin{/_{\mathsf{p}}} PLS_3\rangle) \mathbin{/_{\mathsf{h}}} HLS_2\rangle) \mathbin{/_{\mathsf{n}}} NLS\ ,\end{aligned}$$

where $P_1, \ldots, P_6$ are basic thread expressions, $TLS_1, \ldots, TLS_6$ are local services for threads, PLS_1, PLS_2, PLS_3 are local services for multi-threads, HLS_1, HLS_2 are local services for hosts, and NLS is a local service for networks. It describes a system that consists of two hosts in one network, where on the first host currently a multi-thread with two threads and a multi-thread with three threads exist concurrently, and on the second host currently a single multi-thread with a single thread exists.

A desirable property of all systems designed according to the schema $\|(NV)$ is laid down in Theorem 5 below. That theorem is phrased in terms of the relation sim (is simulated by) on closed terms of $\mathrm{TA}_{\mathrm{fm}}^{\mathrm{tsc,gt,dp,eh}}$+REC defined inductively by means of the rules in Table 18. This relation can be explained as follows: $p\ \mathsf{sim}\ q$ means that, in any execution environment, q performs the same actions as p, in the same order as p, but q possibly performs additional actions prior to each of those common actions and next to the last of those common actions if their number is finite. Roughly speaking, Theorem 5 states that, if a finite thread that forms part of a system designed according to the schema $\|(NV)$ does not make use of the services that form part of the system, then that thread is simulated by the system. In other words, the thread is not really affected by the system.

Theorem 5 (Simulation) *Let P be a basic thread expression in which all basic actions are from the set $\{f.m \mid f \in \mathcal{F} \setminus \{\mathsf{t},\mathsf{p},\mathsf{h},\mathsf{n}\}, m \in \mathcal{M}\}$ and constants standing for the solutions of guarded systems of recursion equations do not occur. Let $C[P]$ be a context of P of the form $\|(NV)$ where NV is a network behaviour vector as above. Then $P\ \mathsf{sim}\ C[P]$. This implies that $C[P]$ will perform all steps of P in finite time.*

Proof We prove this theorem for a more general schema than the schema $\|(NV)$ presented above. We consider the schema that is obtained from the one presented above by replacing all expressions of the form $\|(V)$, where V is a thread vector, a multi-thread vector, a host behaviour vector or a network behaviour vector, by expressions of the form $\mathsf{S}_{\mathsf{D}}^{n}(\|(V))$. Here, for each term p and each $n \geq 0$, the term $\mathsf{S}_{\mathsf{D}}^{n}(p)$ is defined by induction on n as follows: $\mathsf{S}_{\mathsf{D}}^{0}(p)$ is p and $\mathsf{S}_{\mathsf{D}}^{n+1}(p)$ is $\mathsf{S}_{\mathsf{D}}(\mathsf{S}_{\mathsf{D}}^{n}(p))$. The less general schema is covered because $\mathsf{S}_{\mathsf{D}}^{0}(\|(V))$ is $\|(V)$.

Let $TV = \langle P_1 \mathbin{/_{\mathsf{t}}} TLS_1 \rangle \curvearrowright \ldots \curvearrowright \langle P_{l_t} \mathbin{/_{\mathsf{t}}} TLS_{l_t} \rangle$,
$PV = \langle \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV_1)) \mathbin{/_{\mathsf{p}}} PLS_1 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n_{l_p}}(\|(TV_{l_p})) \mathbin{/_{\mathsf{p}}} PLS_{l_p} \rangle$,
$HV = \langle \mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV_1)) \mathbin{/_{\mathsf{h}}} HLS_1 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n'_{l_h}}(\|(PV_{l_h})) \mathbin{/_{\mathsf{h}}} HLS_{l_h} \rangle$,
$NV = \langle \mathsf{S}_{\mathsf{D}}^{n''_1}(\|(HV_1)) \mathbin{/_{\mathsf{n}}} NLS_1 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n''_{l_n}}(\|(HV_{l_n})) \mathbin{/_{\mathsf{n}}} NLS_{l_n} \rangle$
be the thread vector in which P occurs, the multi-thread vector in which TV occurs, the host behaviour vector in which PV occurs and the network behaviour vector in which HV occurs, respectively. Let i_t be the position of P in TV, i_p be the position of TV in PV, i_h be the position of PV in HV, and i_n be the position of HV in NV. Then the *position* of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV))$ is $i_t + l_t(i_p - 1 + l_p(i_h - 1 + l_h(i_n - 1)))$.

We prove $P \;\mathsf{sim}\; C[P]$ by induction on the depth of P and case distinction on the structure of P:

- $P \equiv \mathsf{S}$: $\mathsf{S} \;\mathsf{sim}\; C[\mathsf{S}]$ follows immediately from the definition of sim;
- $P \equiv \mathsf{D}$: $\mathsf{D} \;\mathsf{sim}\; C[\mathsf{D}]$ follows immediately from the definition of sim;
- $P \equiv P' \trianglelefteq f.m \trianglerighteq P''$:
  We prove this case by induction on the position of P in $\|(NV)$:
  - Position of P in $\|(NV)$ is 1:
    Because $P_1 \equiv P' \trianglelefteq f.m \trianglerighteq P''$, we derive, using TSC4, CSI5 and S2D4, $\mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV)) = \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV')) \trianglelefteq f.m \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV''))$ (1), where
    $TV' = \langle P_2 \mathbin{/_{\mathsf{t}}} TLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle P_{l_t} \mathbin{/_{\mathsf{t}}} TLS_{l_t} \rangle \curvearrowright \langle P' \mathbin{/_{\mathsf{t}}} TLS_1 \rangle$,
    $TV'' = \langle P_2 \mathbin{/_{\mathsf{t}}} TLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle P_{l_t} \mathbin{/_{\mathsf{t}}} TLS_{l_t} \rangle \curvearrowright \langle P'' \mathbin{/_{\mathsf{t}}} TLS_1 \rangle$.
    Because $TV_1 \equiv TV$, we derive from (1), using TSC4, CSI5 and S2D4, $\mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV)) = \mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV')) \trianglelefteq f.m \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV''))$ (2), where
    $PV' = \langle \mathsf{S}_{\mathsf{D}}^{n_2}(\|(TV_2)) \mathbin{/_{\mathsf{p}}} PLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n_{l_p}}(\|(TV_{l_p})) \mathbin{/_{\mathsf{p}}} PLS_{l_p} \rangle$
    $\curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV')) \mathbin{/_{\mathsf{p}}} PLS_1 \rangle$,
    $PV'' = \langle \mathsf{S}_{\mathsf{D}}^{n_2}(\|(TV_2)) \mathbin{/_{\mathsf{p}}} PLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n_{l_p}}(\|(TV_{l_p})) \mathbin{/_{\mathsf{p}}} PLS_{l_p} \rangle$
    $\curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV'')) \mathbin{/_{\mathsf{p}}} PLS_1 \rangle$.
    Because $PV_1 \equiv PV$, we derive from (2), using TSC4, CSI5 and S2D4, $\mathsf{S}_{\mathsf{D}}^{n''_1}(\|(HV)) = \mathsf{S}_{\mathsf{D}}^{n''_1}(\|(HV')) \trianglelefteq f.m \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n''_1}(\|(HV''))$ (3), where
    $HV' = \langle \mathsf{S}_{\mathsf{D}}^{n'_2}(\|(PV_2)) \mathbin{/_{\mathsf{h}}} HLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n'_{l_h}}(\|(PV_{l_h})) \mathbin{/_{\mathsf{h}}} HLS_{l_h} \rangle$
    $\curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV')) \mathbin{/_{\mathsf{h}}} HLS_1 \rangle$,
    $HV'' = \langle \mathsf{S}_{\mathsf{D}}^{n'_2}(\|(PV_2)) \mathbin{/_{\mathsf{h}}} HLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n'_{l_h}}(\|(PV_{l_h})) \mathbin{/_{\mathsf{h}}} HLS_{l_h} \rangle$
    $\curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV'')) \mathbin{/_{\mathsf{h}}} HLS_1 \rangle$.
    Because $HV_1 \equiv HV$, we derive from (3), using TSC4, CSI5 and S2D4, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{S}_{\mathsf{D}}^{n}(\|(NV')) \trianglelefteq f.m \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n}(\|(NV''))$ (4), where
    $NV' = \langle \mathsf{S}_{\mathsf{D}}^{n''_2}(\|(HV_2)) \mathbin{/_{\mathsf{n}}} NLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n''_{l_n}}(\|(HV_{l_n})) \mathbin{/_{\mathsf{n}}} NLS_{l_n} \rangle$
    $\curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n''_1}(\|(HV')) \mathbin{/_{\mathsf{n}}} NLS_1 \rangle$,
    $NV'' = \langle \mathsf{S}_{\mathsf{D}}^{n''_2}(\|(HV_2)) \mathbin{/_{\mathsf{n}}} NLS_2 \rangle \curvearrowright \ldots \curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n''_{l_n}}(\|(HV_{l_n})) \mathbin{/_{\mathsf{n}}} NLS_{l_n} \rangle$
    $\curvearrowright \langle \mathsf{S}_{\mathsf{D}}^{n''_1}(\|(HV'')) \mathbin{/_{\mathsf{n}}} NLS_1 \rangle$.
    The depth of NV' and NV'' is one less than the depth of NV. Hence, it follows from (4), using the induction hypothesis and the definition of sim, that $P' \trianglelefteq f.m \trianglerighteq P'' \;\mathsf{sim}\; C[P' \trianglelefteq f.m \trianglerighteq P'']$.

Below, in similar pieces of proof, more than one case must be considered because TSC4, TSC5, TSC6 or TSC7 is applicable where above only TSC4 is applicable.

– Position of P in $\|(NV)$ is greater than 1:
Let $TV_1 = \langle P_1 /_{\mathsf{t}}\ TLS_1\rangle \frown \ldots \frown \langle P_{l_t} /_{\mathsf{t}}\ TLS_{l_t}\rangle$,
$PV_1 = \langle \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV_1)) /_{\mathsf{p}}\ PLS_1\rangle \frown \ldots \frown \langle \mathsf{S}_{\mathsf{D}}^{n_{l_p}}(\|(TV_{l_p})) /_{\mathsf{p}}\ PLS_{l_p}\rangle$,
$HV_1 = \langle \mathsf{S}_{\mathsf{D}}^{n'_1}(\|(PV_1)) /_{\mathsf{h}}\ HLS_1\rangle \frown \ldots \frown \langle \mathsf{S}_{\mathsf{D}}^{n'_{l_h}}(\|(PV_{l_h})) /_{\mathsf{h}}\ HLS_{l_h}\rangle$
be the thread vector at position 1 in PV_1, the multi-thread vector at position 1 in HV_1 and the host behaviour vector at position 1 in NV, respectively. We make a case distinction on the structure of P_1:
  - $P_1 \equiv \mathsf{S}$: We derive, using TSC1 and CSI2, $\mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV_1)) = \mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV'_1))$, where $TV'_1 = \langle P_2 /_{\mathsf{t}}\ TLS_2\rangle \frown \ldots \frown \langle P_{l_t} /_{\mathsf{t}}\ TLS_{l_t}\rangle$. Therefore, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{S}_{\mathsf{D}}^{n}(\|(NV'))$, where NV' is NV with $\mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV_1))$ replaced by $\mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV'_1))$. The position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV'))$ is one less than the position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV))$. Hence, it follows, using the induction hypothesis, that $P\ \mathsf{sim}\ C[P]$.
  - $P_1 \equiv \mathsf{D}$: We derive, using TSC2 and CSI3, $\mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV_1)) = \mathsf{S}_{\mathsf{D}}^{n_1+1}(\|(TV'_1))$, where $TV'_1 = \langle P_2 /_{\mathsf{t}}\ TLS_2\rangle \frown \ldots \frown \langle P_{l_t} /_{\mathsf{t}}\ TLS_{l_t}\rangle$. Therefore, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{S}_{\mathsf{D}}^{n}(\|(NV'))$, where NV' is NV with $\mathsf{S}_{\mathsf{D}}^{n_1}(\|(TV_1))$ replaced by $\mathsf{S}_{\mathsf{D}}^{n_1+1}(\|(TV'_1))$. The position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV'))$ is one less than the position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV))$. Hence, it follows, using the induction hypothesis, that $P\ \mathsf{sim}\ C[P]$.
  - $P_1 \equiv P'_1 \trianglelefteq f_1.m_1 \trianglerighteq P''_1$: On similar lines as for P above, we derive, using TSC2–TSC7, CSI3–CSI5 and S2D3–S2D4, either $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{S}_{\mathsf{D}}^{n}(\|(NV')) \trianglelefteq f_1.m_1 \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n}(\|(NV''))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{tau} \circ \mathsf{S}_{\mathsf{D}}^{n}(\|(NV^*))$ or $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{S}_{\mathsf{D}}^{n+1}(\|(NV^{**}))$, where NV', NV'', NV^* and NV^{**} are such that the position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV'))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV''))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV^*))$ and $\mathsf{S}_{\mathsf{D}}^{n+1}(\|(NV^{**}))$ is one less than the position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV))$. In each case, it follows, using the induction hypothesis and the definition of sim, that $P\ \mathsf{sim}\ C[P]$.
  - $P_1 \equiv P'_1 \trianglelefteq f_1!m_1 \trianglerighteq P''_1$: On similar lines as for P above, we derive, using TSC2–TSC11, CSI3–CSI6 and S2D3–S2D5, either $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = (\mathsf{S}_{\mathsf{D}}^{n}(\|(NV')) \trianglelefteq f_1.m_1 \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n}(\|(NV''))) \trianglelefteq f_1?m_1 \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n}(\|(NV'''))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{S}_{\mathsf{D}}^{n}(\|(NV^*))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{tau} \circ \mathsf{S}_{\mathsf{D}}^{n}(\|(NV^{**}))$ or $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{S}_{\mathsf{D}}^{n+1}(\|(NV^{***}))$, where NV', NV'', $\ldots$ are such that the position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV'))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV''))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV'''))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV^*))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV^{**}))$ and $\mathsf{S}_{\mathsf{D}}^{n+1}(\|(NV^{***}))$ is one less than the position of P in $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV))$. In each case, it follows, using the induction hypothesis and the definition of sim, that $P\ \mathsf{sim}\ C[P]$.
  - $P_1 \equiv P'_1 \trianglelefteq f_1.m_1\ [P''_1] \trianglerighteq P'''_1$: On similar lines as for P above, we derive, using TSC2–TSC7, TSC12–TSC14, CSI3–CSI5, CSI7, S2D3–S2D4 and S2D6, either $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = (\mathsf{S}_{\mathsf{D}}^{n}(\|(NV')) \trianglelefteq f_1.m_1 \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n}(\|(NV''))) \trianglelefteq f_1??m_1 \trianglerighteq \mathsf{S}_{\mathsf{D}}^{n}(\|(NV'''))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{S}_{\mathsf{D}}^{n}(\|(NV^*))$, $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) = \mathsf{tau} \circ \mathsf{S}_{\mathsf{D}}^{n+1}(\|(NV^{**}))$ or $\mathsf{S}_{\mathsf{D}}^{n}(\|(NV)) =$

$\mathsf{tau} \circ \mathsf{S}^n_\mathsf{D}(\|(NV^{***}))$, where NV', NV'', $\ldots$ are such that the position of P in $\mathsf{S}^n_\mathsf{D}(\|(NV'))$, $\mathsf{S}^n_\mathsf{D}(\|(NV''))$, $\mathsf{S}^n_\mathsf{D}(\|(NV'''))$, $\mathsf{S}^n_\mathsf{D}(\|(NV^{*}))$, $\mathsf{S}^{n+1}_\mathsf{D}(\|(NV^{**}))$ and $\mathsf{S}^n_\mathsf{D}(\|(NV^{***}))$ is one less than the position of P in $\mathsf{S}^n_\mathsf{D}(\|(NV))$. In each case, it follows, using the induction hypothesis and the definition of sim, that $P \ \mathsf{sim} \ C[P]$.

- $P_1 \equiv P_1' \trianglelefteq f_1!m_1\,[P_1''] \trianglerighteq P_1'''$: On similar lines as for P above, we derive, using TSC2–TSC14, CSI3–CSI7 and S2D3–S2D6, either $\mathsf{S}^n_\mathsf{D}(\|(NV)) = ((\mathsf{S}^n_\mathsf{D}(\|(NV')) \trianglelefteq f_1.m_1 \trianglerighteq \mathsf{S}^n_\mathsf{D}(\|(NV''))) \trianglelefteq f_1?m_1 \trianglerighteq \mathsf{S}^n_\mathsf{D}(\|(NV'''))) \trianglelefteq f_1??m_1 \trianglerighteq \mathsf{S}^n_\mathsf{D}(\|(NV''''))$, $\mathsf{S}^n_\mathsf{D}(\|(NV)) = \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{S}^n_\mathsf{D}(\|(NV^{*}))$, $\mathsf{S}^n_\mathsf{D}(\|(NV)) = \mathsf{tau} \circ \mathsf{tau} \circ \mathsf{S}^n_\mathsf{D}(\|(NV^{**}))$, $\mathsf{S}^n_\mathsf{D}(\|(NV)) = \mathsf{tau} \circ \mathsf{S}^{n+1}_\mathsf{D}(\|(NV^{***}))$ or $\mathsf{S}^n_\mathsf{D}(\|(NV)) = \mathsf{tau} \circ \mathsf{S}^n_\mathsf{D}(\|(NV^{****}))$, where NV', NV'', $\ldots$ are such that the position of P in $\mathsf{S}^n_\mathsf{D}(\|(NV'))$, $\mathsf{S}^n_\mathsf{D}(\|(NV''))$, $\mathsf{S}^n_\mathsf{D}(\|(NV'''))$, $\mathsf{S}^n_\mathsf{D}(\|(NV''''))$, $\mathsf{S}^n_\mathsf{D}(\|(NV^{*}))$, $\mathsf{S}^{n+1}_\mathsf{D}(\|(NV^{**}))$, $\mathsf{S}^n_\mathsf{D}(\|(NV^{***}))$ and $\mathsf{S}^n_\mathsf{D}(\|(NV^{****}))$ is one less than the position of P in $\mathsf{S}^n_\mathsf{D}(\|(NV))$. In each case, it follows, using the induction hypothesis and the definition of sim, that $P \ \mathsf{sim} \ C[P]$.
- $P_1 \equiv \langle X|E\rangle$: Let t_X be the right hand side of the equation for X in E. By RDP, $\langle X|E\rangle = \langle t_X|E\rangle$. Hence, in this case P_1 can be replaced by $\langle t_X|E\rangle$. The structure of $\langle t_X|E\rangle$ is covered by one of the previous cases.

□

In the proof of $P \ \mathsf{sim} \ C[P]$ for the case $P \equiv P' \trianglelefteq f.m \trianglerighteq P''$ given above, we show among other things that multi-level cyclic interleaving (in the presence of delayed processing and exception handling) is fair, i.e. that there will always come a next turn for all active threads, multi-threads, etc. For the single-level case, a mathematically precise definition of a fair interleaving strategy is given in [8].

8 Conclusions

We have presented an algebraic theory of threads and multi-threading based on multi-level strategic interleaving for the simple strategy of cyclic interleaving. The other interleaving strategies treated in [7] can be adapted to the setting of multi-level strategic interleaving in a similar way. We have also presented a reasonable though simplified formal representation schema of the design of systems that consist of several multi-threaded programs on various hosts in different networks. By dealing with delays and exceptions, this schema is sufficiently expressive to formalize mechanisms like Java pipes (for communication between threads) and Unix sockets (for communication between multi-threads, called processes in Unix jargon, and communication between hosts). The exception handling notation introduced is only used for single threads.

To the best of our knowledge, there is no other work on the theory of threads and multi-threading that is based on strategic interleaving. Although a deterministic interleaving strategy is always used for thread interleaving, it is the practice in work in which the semantics of multi-threated programs is involved to look upon thread interleaving as arbitrary interleaving, see e.g. [1,9].

Options for future work include:

- formalization of mechanisms like Java pipes and Unix sockets using the thread algebra developed in this paper;
- adaptation of some interleaving strategies from [7], other than cyclic interleaving, to the setting of multi-level strategic interleaving;
- extension of the program algebra from [6] with features for delayed processing and exception handling, with a behavioural semantics based on the thread algebra developed in this paper.

Acknowledgements We thank Mark van der Zwaag from the University of Amsterdam, Programming Research Group, for suggesting a substantial improvement of the structural operational semantics of $\mathrm{TA}_{\mathrm{fm}}$ presented in a draft of this paper, and for suggesting the use of the symbol $\updownarrow$ to denote the auxiliary transition relation employed in the resulting structural operational semantics [13]. We also thank an anonymous referee for his/her valuable comments concerning the presentation of the paper.

References

1. E. Ábrahám, F. S. de Boer, W. P. de Roever, and M. Steffen. A compositional operational semantics for JavaMT. In N. Dershowitz, editor, *Verification: Theory and Practice*, volume 2772 of *Lecture Notes in Computer Science*, pages 290–303. Springer-Verlag, 2003.
2. L. Aceto, W. J. Fokkink, and C. Verhoef. Structural operational semantics. In J. A. Bergstra, A. Ponse, and S. A. Smolka, editors, *Handbook of Process Algebra*, pages 197–292. Elsevier, Amsterdam, 2001.
3. J. A. Bergstra and I. Bethke. Polarized process algebra and program equivalence. In J. C. M. Baeten, J. K. Lenstra, J. Parrow, and G. J. Woeginger, editors, *Proceedings 30th ICALP*, volume 2719 of *Lecture Notes in Computer Science*, pages 1–21. Springer-Verlag, 2003.
4. J. A. Bergstra and I. Bethke. Polarized process algebra with reactive composition. *Theoretical Computer Science*, 343:285–304, 2005.
5. J. A. Bergstra and J. W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1/3):109–137, 1984.
6. J. A. Bergstra and M. E. Loots. Program algebra for sequential code. *Journal of Logic and Algebraic Programming*, 51(2):125–156, 2002.
7. J. A. Bergstra and C. A. Middelburg. Thread algebra for strategic interleaving. Computer Science Report 04-35, Department of Mathematics and Computer Science, Eindhoven University of Technology, November 2004.
8. J. A. Bergstra and C. A. Middelburg. Simulating Turing machines on Maurer machines. Computer Science Report 05-28, Department of Mathematics and Computer Science, Eindhoven University of Technology, November 2005.
9. C. Flanagan, S. N. Freund, S. Qadeer, and S. A. Seshia. Modular verification of multi-threaded programs. *Theoretical Computer Science*, 338(1/3):153–183, 2005.
10. J. Gosling, B. Joy, G. Steele, and G. Bracha. *The Java Language Specification*. Addison-Wesley, Reading, MA, second edition, 2000.

11. A. Hejlsberg, S. Wiltamuth, and P. Golde. *C# Language Specification*. Addison-Wesley, Reading, MA, 2003.
12. C. A. Middelburg. An alternative formulation of operational conservativity with binding terms. *Journal of Logic and Algebraic Programming*, 55(1/2):1–19, 2003.
13. M. B. van der Zwaag. Personal communication, 2006.