Modal logics for Brane Calculus

Modal logics for Brane Calculus

Marino Miculan Giorgio Bacci

Dept. of Mathematics and Computer Science
University of Udine, Italy. mm@uniud.it

Abstract. The Brane Calculus is a calculus of mobile processes, intended to model the transport machinery of a cell system. In this paper, we introduce the *Brane Logic*, a modal logic for expressing formally properties about systems in Brane Calculus. Similarly to previous logics for mobile ambients, Brane Logic has specific spatial and temporal modalities. Moreover, since in Brane Calculus the activity resides on membrane surfaces and not inside membranes, we need to add a specific logic (akin Hennessy-Milner's) for reasoning about membrane activity.
We present also a proof system for deriving valid sequents in Brane Logic. Finally, we present a model checker for a decidable fragment of this logic.

1 Introduction

In [4], Cardelli has proposed a schematic model of biological systems as three different and interacting abstract machines. Following the approach pioneered in [13], these abstract machines are modelled using methodologies borrowed from the theory of concurrent systems.

The most abstract of these three machines is the *membrane machine*, which focuses on the dynamics of biological membranes. At this level of abstraction, a biological system is seen as a hierarchy of compartments, which can interact by changing their position. In order to model this machinery, Cardelli has introduced the *Brane Calculus* [3], a calculus of mobile nested processes where the computational activity takes place *on* membranes, not inside them. A process of this represents a system of nested membranes; the evolution of a process corresponds to membrane interactions (phagocytosis, endo/exocytosis, ...).

Having such a formal representation of the membrane machine, a natural question is how to express formally also the *biological properties*, that is, the "statements" about a given system. Some examples are the following:

> "If a macrophage is exposed to target cells that have been evenly coated with antibody, it ingests the coated cells." [1, Chap.6, p.335]
> "The [...] Rous sarcoma virus [...] can transform a cell into a cancer cell." [1, Chap.8, p.417]
> "The virus escapes from the endosome" [1, Chap.8, p.469]

In our opinion, it is highly desirable to be able to express formally (i.e., in a well-specified logical formalism) this kind of properties. First, this would avoid the intrinsic ambiguity of natural language, ruling out any misinterpretation of

the meaning of a statement. Secondly, such a logical formalism can be used for defining *specifications of systems*, i.e. requirements that a system must satisfy. These specifications can be used in *(semi)automatic verification* of existing systems (using model-checking or static analysis techniques), or in *(semi)automatic synthesis* of new systems (meeting the given specification). Finally, the logical formalism yields naturally a formal notion of *system equivalence:* two systems are equivalent if they satisfy precisely the same properties. Often this equivalence implies observational equivalence (depending on the expressive power of the logical formalism), so a subsystem can be replaced with a logically equivalent one (possibly synthetic) without altering the behaviour of the whole system.

The aim of this work is to take a step in this direction. We introduce the *Brane Logic*, a modal logic specifically designed for expressing properties about systems described using the Brane Calculus. Modal logics are commonly used in concurrency theory for describing behaviour of concurrent systems. In particular, we take inspiration from Ambient Logic, the logic for Ambient calculus [5]. Like Ambient Logic, our logic features *spatial* and *temporal* modalities, which are specific logical operators for expressing properties about the topology and the dynamic behaviour of nested systems. However, differently from Ambient Logic, we need to define also a specific logic for expressing properties of membranes themselves. Each membrane can be seen as a flat surface where different agents can interact, but without nestings. Thus membranes are more similar to CCS than to Ambients; as a consequence, the logic for membranes is similar to Hennessy-Milner's logic [8], extended with spatial connectives as in [2].

After having defined Brane Logic and its formal interpretation over the Brane Calculus (Section 3), in Section 4 we consider *sequents*, and introduce a set of valid *inference rules* (with many derivable corollaries). Several examples throughout the paper will illustrate the expressive power of the logic. Finally, in Section 5, we single out a fragment of the calculus and of the logic for which the satisfiability problem is decidable and for which we give a model checker algorithm. Conclusions, final remarks and directions for future work are in Section 6.

2 Summary of Brane calculus

In this paper we focus on the basic version of Brane Calculus without communication primitives and molecular complexes. For a description of the intuitive meaning of the language and the reduction rules, we refer the reader to [3].

Syntax of (Basic) Brane Calculus

Systems Π :	$P, Q ::= \diamond \mid \sigma\{\!\!\{P\}\!\!\} \mid P \circ Q \mid !P$
Membranes Σ :	$\sigma, \tau ::= \mathbf{0} \mid \sigma \vert \tau \mid a.\sigma \mid !\sigma$
Actions Ξ :	$a, b ::= \mathfrak{d}_n \mid \mathfrak{d}_n^{\perp}(\sigma) \mid \mathfrak{D}_n \mid \mathfrak{D}_n^{\perp} \mid \circledcirc(\sigma)$

where n is taken from a countable set Λ of *names*. We will write a, $\{\!\!\{P\}\!\!\}$ and $\sigma\{\!\!\}$, instead of $a.\mathbf{0}$, $\mathbf{0}\{\!\!\{P\}\!\!\}$ and $\sigma\{\!\!\{\diamond\}\!\!\}$, respectively.

The set of free names of a system P, of a membrane σ and of an action a, denoted by $\mathsf{FN}(P)$, $\mathsf{FN}(\sigma)$, $\mathsf{FN}(a)$ respectively, are defined as usual; notice that in this syntax there are no binders.

As in many process calculi, terms of the Brane Calculus can be rearranged according to a structural congruence relation ($\equiv$). For a formal definition see [3].

The dynamic behaviour of Brane Calculus is specified by means of a reduction relation ("reaction") between systems $P \rightarrow Q$, whose rules are the following:

Operational Semantics

$$\mathfrak{d}_n^{\perp}(\rho).\tau|\tau_0\{\!|Q|\!\} \circ \mathfrak{d}_n.\sigma|\sigma_0\{\!|P|\!\} \rightarrow \tau|\tau_0\{\!|\rho\{\!|\sigma|\sigma_0\{\!|P|\!\}\!|\!\} \circ Q|\!\} \quad \text{(React phago)}$$

$$\mathfrak{d}_n^{\perp}.\tau|\tau_0\{\!|\mathfrak{d}_n.\sigma|\sigma_0\{\!|P|\!\} \circ Q|\!\} \rightarrow \sigma|\sigma_0|\tau|\tau_0\{\!|Q|\!\} \circ P \quad \text{(React exo)}$$

$$\circledcirc(\rho).\sigma|\sigma_0\{\!|P|\!\} \rightarrow \sigma|\sigma_0\{\!|\rho\{\!|\diamond|\!\} \circ P|\!\} \quad \text{(React pino)}$$

$$\frac{P \rightarrow Q}{\sigma\{\!|P|\!\} \rightarrow \sigma\{\!|Q|\!\}} \qquad \frac{P \rightarrow Q}{P \circ R \rightarrow Q \circ R} \quad \text{(React loc, React comp)}$$

$$\frac{P \equiv P' \quad P' \rightarrow Q' \quad Q' \equiv Q}{P \rightarrow Q} \quad \text{(React equiv)}$$

We denote by $\rightarrow^*$ the usual reflexive and transitive closure of $\rightarrow$.

As in [3], the Mate-Bud-Drip calculus is easily encoded, as follows:

Derived membrane constructors and reaction

Mate : $\mathsf{mate}_n.\sigma \triangleq \mathfrak{d}_n.\mathfrak{d}_{n'}.\sigma \qquad \mathsf{mate}_n^{\perp}.\tau \triangleq \mathfrak{d}_n^{\perp}(\mathfrak{d}_{n'}^{\perp}.\mathfrak{d}_{n''}).\mathfrak{d}_{n''}^{\perp}.\tau$
$\mathsf{mate}_n.\sigma|\sigma_0\{\!|P|\!\} \circ \mathsf{mate}_n^{\perp}.\tau|\tau_0\{\!|Q|\!\} \rightarrow^* \sigma|\sigma_0|\tau|\tau_0\{\!|P \circ Q|\!\}$

Bud : $\mathsf{bud}_n.\sigma \triangleq \mathfrak{d}_n.\sigma \qquad \mathsf{bud}_n^{\perp}(\rho).\tau \triangleq \circledcirc(\mathfrak{d}_n^{\perp}(\rho).\mathfrak{d}_{n'}).\mathfrak{d}_{n'}^{\perp}.\tau$
$\mathsf{bud}_n^{\perp}(\rho).\tau|\tau_0\{\!|\mathsf{bud}_n.\sigma|\sigma_0\{\!|P|\!\} \circ Q|\!\} \rightarrow^* \rho\{\!|\sigma|\sigma_0\{\!|P|\!\}\!|\!\} \circ \tau|\tau_0\{\!|Q|\!\}$

Drip : $\mathsf{drip}_n.(\rho).\sigma \triangleq \circledcirc(\circledcirc(\rho).\mathfrak{d}_n).\mathfrak{d}_n^{\perp}.\sigma$
$\mathsf{drip}_n(\rho).\sigma|\sigma_0\{\!|P|\!\} \rightarrow^* \rho\{\!|\diamond|\!\} \circ \sigma|\sigma_0\{\!|P|\!\}$

3 The Brane Logic

In this section we introduce a logic for expressing properties of systems of the Brane Calculus, called *Brane Logic*. Like similar temporal-spatial logics, such as Ambient Logic [5] and Separation Logic [14], Brane Logic features special modal connectives for expressing spatial properties (i.e., about relative positions) and behavioural properties. The main difference between its closest ancestor (Ambient Logic), is that Brane Logic can express properties about the actions which can take place *on membranes*, not only in systems. Thus, there are actually two spatial logics, interacting each other: one for reasoning about membranes (called *membrane logic*) and one for reasoning about systems (the *system logic*).

Syntax The syntax of the Brane Logic is the following:

Syntax of Brane Logic

System formulas Φ

$\mathcal{A}, \mathcal{B} ::= \mathbf{T} \mid \neg\mathcal{A} \mid \mathcal{A} \vee \mathcal{B}$	(classical propositional fragment)
$\diamond$	(void system)
$\mathcal{M}\{\!|\mathcal{A}|\!\} \mid \mathcal{A}@\mathcal{M}$	(compartment, compartment adjoint)
$\mathcal{A} \circ \mathcal{B} \mid \mathcal{A} \rhd \mathcal{B}$	(spatial composition, composition adjoint)
$\Diamond\mathcal{A} \mid \Diamond\!\!\!-\mathcal{A}$	(eventually modality, somewhere modality)
$\forall x.\mathcal{A}$	(quantification over names)

Membrane formulas Ω

$\mathcal{M},\mathcal{N} ::=$	$\mathbf{T} \mid \neg\mathcal{M} \mid \mathcal{M} \vee \mathcal{N}$	(classical propositional fragment)
	$\mathbf{0}$	(void membrane)
	$\mathcal{M}\vert\mathcal{N} \mid \mathcal{M} \blacktriangleright \mathcal{N}$	(spatial composition, composition adjoint)
	$\langle\alpha\rangle\mathcal{M}$	(action modality)

Action formulas Θ

$\alpha,\beta ::=$	$\mathfrak{D}_\eta \mid \mathfrak{D}^\perp_\eta(\mathcal{M})$	(phago, co-phago)
	$\mathfrak{O}_\eta \mid \mathfrak{O}^\perp_\eta$	(exo, co-exo)
	$\circledcirc(\mathcal{M})$	(pino)
$\eta ::=$	$n \mid x$	(terms)

Given a formula $\mathcal{A}$, its free names $\mathsf{FN}(\mathcal{A})$ are easily defined, since there are no binders for names. Similarly, we can define the set of free variables $\mathsf{FV}(\mathcal{A})$, noticing that the only binder for variables is the universal quantifier. As usual, a formula $\mathcal{A}$ is *closed* if $\mathsf{FV}(\mathcal{A}) = \varnothing$.

For sake of simplicity, we will use the shorthands $\mathcal{M}\{\!\}$ and $\langle\alpha\rangle$ in place of $\mathcal{M}\{\!\diamond\!\}$ and $\langle\alpha\rangle\mathbf{0}$ respectively.

We give next an intuitive explanation of the most unusual constructors.

- P satisfies $\mathcal{M}\{\!\mathcal{A}\!\}$ if $P \equiv \sigma\{\!Q\!\}$, where σ and Q satisfy $\mathcal{M}$ and $\mathcal{A}$ respectively.
- @ e $\triangleright$ are very useful for expressing security and safety properties. A system P satisfies $\mathcal{A}@\mathcal{M}$ if, when P is enclosed in a membrane satisfying $\mathcal{M}$, the resulting system satisfies $\mathcal{A}$. Similarly, a system P satisfies $\mathcal{A} \triangleright \mathcal{B}$ if, when P is put aside a system enjoying $\mathcal{B}$, the whole system satisfies $\mathcal{A}$.
- A membrane σ satisfies $\langle\alpha\rangle\mathcal{M}$ if σ can perform an action satisfying α, yielding a residual satisfying $\mathcal{M}$.
- $\mathcal{M}|\mathcal{N}$ and its adjoint $\mathcal{M} \blacktriangleright \mathcal{N}$ are analogous to $\mathcal{A} \circ \mathcal{B}$ and $\mathcal{A} \triangleright \mathcal{B}$ respectively.

Satisfaction Formally, the meaning of a formula is defined by means of a family of *satisfaction* relations, one for each syntactic sort of logical formulas[1]

$$\models \subseteq \Pi \times \Phi \qquad \models \subseteq \Sigma \times \Omega \qquad \models \subseteq \Xi \times \Theta$$

These relations are defined by induction on the syntax of the formulas. Let us start with satisfaction of systems. First, we have to introduce the *subsystem* relation $P \downarrow Q$ (read "Q is an immediate subsystem of P"), defined as

$$P \downarrow Q \triangleq \exists P' : \Pi, \sigma : \Sigma . P \equiv \sigma\{\!Q\!\}|P'$$

We denote by $\downarrow^*$ the reflexive-transitive closure of $\downarrow$.

Then, we can define the satisfaction of system formulas.

Satisfaction of System Formulas

$\forall P : \Pi$	$P \models \mathbf{T}$	
$\forall P : \Pi, \mathcal{A} : \Phi$	$P \models \neg\mathcal{A}$	$\triangleq P \not\models \mathcal{A}$
$\forall P : \Pi, \mathcal{A}, \mathcal{B} : \Phi$	$P \models \mathcal{A} \vee \mathcal{B}$	$\triangleq P \models \mathcal{A} \vee P \models \mathcal{B}$
$\forall P : \Pi$	$P \models \diamond$	$\triangleq P \equiv \diamond$
$\forall P : \Pi, \mathcal{A} : \Phi, \mathcal{M} : \Omega$	$P \models \mathcal{M}\{\!\mathcal{A}\!\}$	$\triangleq \exists P' : \Pi, \sigma : \Sigma . P \equiv \sigma\{\!P'\!\} \wedge P' \models \mathcal{A} \wedge \sigma \models \mathcal{M}$

[1] We will use the same symbol $\models$ for the three relations, since they are easily distinguishable from the context.

$$
\begin{array}{lll}
\forall P : \Pi, \mathcal{A}, \mathcal{B} : \Phi & P \vDash \mathcal{A} \circ \mathcal{B} & \triangleq \exists P', P'' : \Pi . P \equiv P' \circ P'' \wedge P' \vDash \mathcal{A} \wedge P'' \vDash \mathcal{B} \\
\forall P : \Pi, \mathcal{A} : \Phi, x : \vartheta & P \vDash \forall x . \mathcal{A} & \triangleq \forall m : \Lambda . P \vDash \mathcal{A}\{x \leftarrow m\} \\
\forall P : \Pi, \mathcal{A} : \Phi & P \vDash \Diamond \mathcal{A} & \triangleq \exists P' : \Pi . P \blacktriangleright^* P' \wedge P' \vDash \mathcal{A} \\
\forall P : \Pi, \mathcal{A} : \Phi & P \vDash \Diamond\!\!\!\!- \mathcal{A} & \triangleq \exists P' : \Pi . P \downarrow^* P' \wedge P' \vDash \mathcal{A} \\
\forall P : \Pi, \mathcal{A} : \Phi, \mathcal{M} : \Omega & P \vDash \mathcal{A} @ \mathcal{M} & \triangleq \forall \sigma : \Sigma . \sigma \vDash \mathcal{M} \Rightarrow \sigma \langle\!| P |\!\rangle \vDash \mathcal{A} \\
\forall P : \Pi, \mathcal{A}, \mathcal{B} : \Phi & P \vDash \mathcal{A} \rhd \mathcal{B} & \triangleq \forall P' : \Pi . P' \vDash \mathcal{A} \Rightarrow P \circ P' \vDash \mathcal{B}
\end{array}
$$

This definition relies on the satisfaction of membrane formulas, which we define next. To this end, we need to introduce a notion of *membrane observation*, by means of a *labelled transition system* (LTS) $\sigma \xrightarrow{l} \tau$ for membranes. A crucial point is how to define correctly the labels (i.e., the observations) l of this LTS.

The evident similarity between membranes and Milner's CCS [12] could suggest to define observations simply as *actions;* e.g., we could take $a.\sigma \xrightarrow{a} \sigma$. However, an important difference between membranes and CCS is that in latter case, the labels are τ and communications over channels, i.e. names (possibly together with terms, which are separated from processes in any case). On the other hand, actions in membranes form a whole language, which incorporates also the membranes themselves. Thus, observing actions over the membranes would mean to observe explicitly (also) membranes instead of some abstract logical property. For instance, in the transition $\mathfrak{o}(\sigma).\tau \xrightarrow{\mathfrak{o}(\sigma)} \tau$ we have a specific membrane σ in the label. This kind of observation is too "fine-grained" and intensional with respect to the rest of the logic, which never deals with specific membranes but only with their properties.

Therefore, we choose to take as labels the *action formulas*, instead of actions. Thus the LTS is a relation $\sigma \xrightarrow{\alpha} \tau$, which reads as "$\sigma$ performs an action satisfying α, and reduces to τ". This LTS is defined by the following rules:

Labelled Transition System for Membranes

$$
\frac{a \vDash \alpha}{a.\sigma \xrightarrow{\alpha} \sigma}(\text{prefix}) \qquad \frac{\sigma \xrightarrow{\alpha} \sigma'}{\sigma | \tau \xrightarrow{\alpha} \sigma' | \tau}(\text{par}) \qquad \frac{\sigma \equiv \sigma' \quad \sigma' \xrightarrow{\alpha} \tau' \quad \tau' \equiv \tau}{\sigma \xrightarrow{\alpha} \tau}(\text{equiv})
$$

Notice that in the (prefix) rule we use the satisfaction relation for actions:

Satisfaction of action formulas

$$
\begin{array}{lll}
\forall a : \Gamma, n : \Lambda & a \vDash \mathfrak{o}_n & \triangleq a = \mathfrak{o}_n \\
\forall a : \Gamma, n : \Lambda, \mathcal{M} : \Omega & a \vDash \mathfrak{o}_n^{\perp}(\mathcal{M}) & \triangleq \exists \sigma : \Sigma . a = \mathfrak{o}_n^{\perp}(\sigma) \wedge \sigma \vDash \mathcal{M} \\
\forall a : \Gamma, n : \Lambda & a \vDash \mathfrak{O}_n & \triangleq a = \mathfrak{O}_n \\
\forall a : \Gamma, n : \Lambda & a \vDash \mathfrak{O}_n^{\perp} & \triangleq a = \mathfrak{O}_n^{\perp} \\
\forall a : \Gamma, \mathcal{M} : \Omega & a \vDash \circledcirc(\mathcal{M}) & \triangleq \exists \sigma : \Sigma . a = \circledcirc(\sigma) \wedge \sigma \vDash \mathcal{M}
\end{array}
$$

This relation is defined in terms of the satisfaction of membrane formulas:

Satisfaction of membrane formulas

$$
\begin{array}{lll}
\forall \sigma : \Sigma & \sigma \vDash \mathbf{T} & \\
\forall \sigma : \Sigma, \mathcal{M} : \Omega & \sigma \vDash \neg \mathcal{M} & \triangleq \sigma \nvDash \mathcal{M} \\
\forall \sigma : \Sigma, \mathcal{M}, \mathcal{N} : \Omega & \sigma \vDash \mathcal{M} \vee \mathcal{N} & \triangleq \sigma \vDash \mathcal{M} \vee \sigma \vDash \mathcal{M}
\end{array}
$$

$\forall \sigma : \Sigma$	$\sigma \vDash \mathbf{0}$	$\triangleq \sigma \equiv \mathbf{0}$
$\forall \sigma : \Sigma, \mathcal{N}, \mathcal{M} : \Omega$	$\sigma \vDash \mathcal{M}\|\mathcal{N}$	$\triangleq \exists \sigma', \sigma'' : \Sigma . \sigma \equiv \sigma' \| \sigma'' \wedge \sigma' \vDash \mathcal{M} \wedge \sigma'' \vDash \mathcal{N}$
$\forall \sigma : \Sigma, \alpha : \Theta$	$\sigma \vDash \langle \alpha \rangle \mathcal{M}$	$\triangleq \exists \sigma' : \Sigma . \sigma \xrightarrow{\alpha} \sigma' \wedge \sigma' \vDash \mathcal{M}$
$\forall \sigma : \Sigma, \mathcal{M}, \mathcal{N} : \Omega$	$\sigma \vDash \mathcal{M} \blacktriangleright \mathcal{N}$	$\triangleq \forall \sigma' : \Sigma . \sigma' \vDash \mathcal{M} \Rightarrow \sigma \| \sigma' \vDash \mathcal{N}$

Notice that the truth of $\langle \alpha \rangle \mathcal{M}$ is defined using the LTS we defined before. Thus, the LTS, the satisfaction of action formulas, and the satisfaction of membrane formulas are three mutually defined judgments.

Derived connectives In the following table, we introduce several useful derived connectives which can be defined as shorthands of longer formulas, together with an intuitive description of their meaning. This description can be easily checked by unfolding the formal meaning, using the satisfaction relations above.

Some derived connectives

$\mathcal{A} \diamond \mathcal{B} \triangleq \neg(\neg\mathcal{A} \circ \neg\mathcal{B})$	system decomposition
$\mathcal{A}^{\forall} \triangleq \mathcal{A} \diamond \mathbf{F}$	every subsystem (also non proper) satisfies $\mathcal{A}$
$\mathcal{A}^{\exists} \triangleq \mathcal{A} \circ \mathbf{T}$	some subsystem satisfies $\mathcal{A}$
$\mathcal{A} \propto \mathcal{B} \triangleq \neg(\mathcal{B} \rhd \neg\mathcal{A})$	system fusion
$\mathcal{A} \multimap\!\!\Rightarrow \mathcal{B} \triangleq \neg(\mathcal{A} \circ \neg\mathcal{B})$	fusion adjoint
$\mathcal{M} \parallel \mathcal{N} \triangleq \neg(\neg\mathcal{M}\|\neg\mathcal{N})$	membrane decomposition
$\mathcal{M}^{\forall} \triangleq \mathcal{M} \parallel \mathbf{F}$	every part of the membrane satisfies $\mathcal{M}$
$\mathcal{M}^{\exists} \triangleq \mathcal{M}\|\mathbf{T}$	some part of the membrane satisfies $\mathcal{M}$
$\mathcal{M} \ltimes \mathcal{N} \triangleq \neg(\mathcal{N} \blacktriangleright \neg\mathcal{M})$	membrane fusion
$\mathcal{M} \Mapsto \mathcal{N} \triangleq \neg(\mathcal{M}\|\neg\mathcal{N})$	fusion adjoint

Derived connectives for Mate-Bud-Drip

$\langle \mathsf{mate}_\eta \rangle \mathcal{M} \triangleq \langle \mathfrak{O}_\eta \rangle \langle \mathfrak{O}_{\eta'} \rangle \mathcal{M}$	mate
$\langle \mathsf{mate}^\perp_\eta \rangle \mathcal{N} \triangleq \langle \mathfrak{O}^\perp_\eta (\langle \mathfrak{O}^\perp_{\eta'} \rangle \langle \mathfrak{O}_{\eta''} \rangle) \rangle \langle \mathfrak{O}^\perp_{\eta''} \rangle \mathcal{N}$	co-mate
$\langle \mathsf{bud}_\eta \rangle \mathcal{M} \triangleq \langle \mathfrak{O}_\eta \rangle \mathcal{M}$	bud
$\langle \mathsf{bud}^\perp_\eta(\mathcal{K}) \rangle \mathcal{N} \triangleq \langle \circledcirc (\langle \mathfrak{O}^\perp_\eta(\mathcal{K}) \rangle \langle \mathfrak{O}_{\eta'} \rangle) \rangle \langle \mathfrak{O}^\perp_{\eta'} \rangle \mathcal{N}$	co-bud
$\langle \mathsf{drip}_\eta(\mathcal{N}) \rangle \mathcal{M} \triangleq \langle \circledcirc (\langle \circledcirc(\mathcal{N}) \rangle \langle \mathfrak{O}_\eta \rangle) \rangle \langle \mathfrak{O}^\perp_\eta \rangle \mathcal{M}$	drip

Let us describe shortly the meaning of the most important derived connectives; not surprisingly, these are close to similar ones in the Ambient Logic.

System decomposition is the dual of composition, and it is useful to describe invariant properties of systems. A system satisfies $\mathcal{A} \diamond \mathcal{B}$ if, for any decomposition of the system in two parts, a part satisfies $\mathcal{A}$ or the other $\mathcal{B}$. As a consequence, the formula $\mathcal{A}^{\forall}$ means that any decomposition satisfies $\mathcal{A}$, or satisfies $\mathbf{F}$. Since $\mathbf{F}$ is never satisfied, this means that in every possible decomposition, a part satisfies $\mathcal{A}$; hence, every immediate subsystem satisfies $\mathcal{A}$. Thus, the formula

$(\mathcal{M}\{\!|\mathbf{T}|\!\} \Rightarrow \mathcal{M}\{\!|\mathcal{N}\{\!|\mathbf{T}|\!\}|\!\})^{\forall}$ means "every membrane satisfying $\mathcal{M}$ in the system, must contain just a membrane satisfying $\mathcal{N}$".

Dually, $\mathcal{A}^{\exists}$ means that there exists a decomposition of the system where a component satisfies $\mathcal{A}$. Thus, the formula $\mathcal{M}\{\!|\mathcal{N}\{\!|\mathbf{T}|\!\}^{\exists}|\!\}$ states that the system is composed by a membrane satisfying $\mathcal{M}$, which contains at least another membrane satisfying $\mathcal{N}$.

Other interesting applications of derived constructors are, e.g., $\Box\mathcal{M}\{\!|\mathbf{T}|\!\}$ ("the system will be always composed by a single membrane, satisfying $\mathcal{M}$), and $\neg\neg(\mathcal{M}\{\!|\mathbf{T}|\!\}^{\exists})$ ("nowhere there is a membrane satisfying $\mathcal{M}$"). This last formula expresses a *purity* condition (like, e.g., "nowhere there exists a bacterium/virus identified by $\mathcal{M}$", i.e., "the system is free from infections of type $\mathcal{M}$").

The fusion $\mathcal{A} \propto \mathcal{B}$ means that there exists a system satisfying $\mathcal{B}$ such that, when put together with the actual system, the whole system satisfies $\mathcal{A}$. Dually, $\mathcal{A} \multimap\!\!\Rightarrow \mathcal{B}$ means that in any decomposition of the system, whenever a part satisfies $\mathcal{A}$ then the other satisfies $\mathcal{B}$.

We end this section with a basic property of satisfaction relations, that is, that satisfaction is preserved by structural congruence.

Proposition 1 (Satisfaction is up to $\equiv$).
1. $(\sigma \vDash \mathcal{M} \wedge \sigma \equiv \tau) \Rightarrow \tau \vDash \mathcal{M}$ *2.* $(P \vDash \mathcal{A} \wedge P \equiv Q) \Rightarrow Q \vDash \mathcal{A}$

4 Validity and proof system

In this section, we investigate *validity* of formulas or, more generally of *sequents* and *inference rules*. Validity is defined in terms of satisfaction; more precisely, a closed system/membrane/action formula is *valid* if it is satisfied by every system/membrane/action.

4.1 Interpretation of sequents and rules

For sequents and rules we will adopt a notation similar to that of Ambient Logic [5]. A sequent will have exactly one premise and one conclusion, denoted as $\mathcal{A} \vdash \mathcal{B}$; in this way we do not have to decide any (somewhat arbitrary) intrepretation of commas in sequents.

Formally, validity of formulas, sequents and rules is as follows:

Validity of formulas, sequents and rules

$$\mathbf{vld}(\mathcal{A}) \triangleq \forall P : \Pi.P \vDash \mathcal{A} \quad \mathcal{A} \text{ (closed) is valid}$$

$$\mathcal{A} \vdash \mathcal{B} \triangleq \mathbf{vld}(\mathcal{A} \Rightarrow \mathcal{B}) \quad \text{Sequent}$$

$$\mathcal{A} \dashv\vdash \mathcal{B} \triangleq \mathcal{A} \vdash \mathcal{B} \wedge \mathcal{B} \vdash \mathcal{A} \quad \text{Double sequent}$$

$$\frac{\mathcal{A}_1 \vdash \mathcal{B}_1 \cdots \mathcal{A}_n \vdash \mathcal{B}_n}{\mathcal{A}_0 \vdash \mathcal{B}_0} \triangleq \mathcal{A}_1 \vdash \mathcal{B}_1 \wedge \cdots \wedge \mathcal{A}_n \vdash \mathcal{B}_n \Rightarrow \mathcal{A}_0 \vdash \mathcal{B}_0 \quad \text{Inference rule } (n \geq 0)$$

$$\frac{\mathcal{A}_1 \vdash \mathcal{B}_1 \cdots \mathcal{A}_n \vdash \mathcal{B}_n}{\mathcal{A}_0 \dashv\vdash \mathcal{B}_0} \triangleq \mathcal{A}_1 \vdash \mathcal{B}_1 \wedge \cdots \wedge \mathcal{A}_n \vdash \mathcal{B}_n \Rightarrow \mathcal{A}_0 \dashv\vdash \mathcal{B}_0 \quad \text{Double conclusion}$$

$$\frac{\mathcal{A}_1 \vdash \mathcal{B}_1}{\mathcal{A}_2 \vdash \mathcal{B}_2} \triangleq \frac{\mathcal{A}_1 \vdash \mathcal{B}_1}{\mathcal{A}_2 \vdash \mathcal{B}_2} \wedge \frac{\mathcal{A}_2 \vdash \mathcal{B}_2}{\mathcal{A}_1 \vdash \mathcal{B}_1} \quad \text{Double rule}$$

4.2 Logical Rules

In this section we collect several valid sequents and rules for the Brane Logic. We distinguish between "inference rules", which can be seen as proper theorems validated by the interpretation above, and "derived rules", that is corollaries derived by solely applying the inference rules. We omit the rules for propositional calculus which are the same of Ambient Logic [5].

Composition The spatial nature of Brane Logic leads to important rules for reasoning about composition and decomposition of systems and membranes.

Rules for composition of systems and membranes

$$(\circ\diamond)\ \frac{}{\mathcal{A}\circ\diamond \dashv\vdash \mathcal{A}} \qquad (\circ\neg\diamond)\ \frac{}{\mathcal{A}\circ\neg\diamond \vdash \neg\diamond}$$

$$(A\circ)\ \frac{}{\mathcal{A}\circ(\mathcal{B}\circ\mathcal{C}) \dashv\vdash (\mathcal{A}\circ\mathcal{B})\circ\mathcal{C}} \qquad (X\circ)\ \frac{}{\mathcal{A}\circ\mathcal{B}\vdash\mathcal{B}\circ\mathcal{A}}$$

$$(\circ\vee)\ \frac{}{(\mathcal{A}\vee\mathcal{B})\circ\mathcal{C}\vdash\mathcal{A}\circ\mathcal{C}\vee\mathcal{B}\circ\mathcal{C}} \qquad (\circ\vdash)\ \frac{\mathcal{A}'\vdash\mathcal{B}' \quad \mathcal{A}''\vdash\mathcal{B}''}{\mathcal{A}'\circ\mathcal{A}''\vdash\mathcal{B}'\circ\mathcal{B}''}$$

$$(\circ\phi)\ \frac{}{\mathcal{A}'\circ\mathcal{A}''\vdash(\mathcal{A}'\circ\mathcal{B}'')\vee(\mathcal{B}'\circ\mathcal{A}'')\vee(\neg\mathcal{B}'\circ\neg\mathcal{B}'')} \qquad (\circ\rhd)\ \frac{\mathcal{A}\circ\mathcal{C}\vdash\mathcal{B}}{\mathcal{A}\vdash\mathcal{C}\rhd\mathcal{B}}$$

$$(|\mathbf{0})\ \frac{}{\mathcal{M}|\mathbf{0}\dashv\vdash\mathcal{M}} \qquad (|\neg\mathbf{0})\ \frac{}{\mathcal{M}|\neg\mathbf{0}\vdash\neg\mathbf{0}}$$

$$(A|)\ \frac{}{\mathcal{M}|(\mathcal{N}|\mathcal{K})\dashv\vdash(\mathcal{M}|\mathcal{N})|\mathcal{K}} \qquad (X|)\ \frac{}{\mathcal{M}|\mathcal{N}\vdash\mathcal{N}|\mathcal{M}}$$

$$(|\vee)\ \frac{}{(\mathcal{M}\vee\mathcal{N})|\mathcal{K}\vdash\mathcal{M}|\mathcal{K}\vee\mathcal{N}|\mathcal{K}} \qquad (|\vdash)\ \frac{\mathcal{M}'\vdash\mathcal{N}' \quad \mathcal{M}''\vdash\mathcal{N}''}{\mathcal{M}'|\mathcal{M}''\vdash\mathcal{N}'|\mathcal{N}''}$$

$$(|\parallel)\ \frac{}{\mathcal{M}'|\mathcal{M}''\vdash(\mathcal{M}'|\mathcal{N}'')\vee(\mathcal{N}'|\mathcal{M}'')\vee(\neg\mathcal{N}'|\neg\mathcal{N}'')} \qquad (|\blacktriangleright)\ \frac{\mathcal{M}|\mathcal{K}\vdash\mathcal{N}}{\mathcal{M}\vdash\mathcal{K}\blacktriangleright\mathcal{N}}$$

Most of these rules have a direct and intuitive meaning. For instance, $\circ\diamond$ and $\circ\neg\diamond$ state that $\diamond$ is part of any system, and if a part of a system is not void then the whole system is not void. Notice that rule $(\circ\rhd)$ states that $\circ$ is the left adjoint of $\rhd$, as expected; similarly for $|$ and $\blacktriangleright$.

Due to lack of space we cannot show many interesting corollaries; see [11].

Compartments The rules for reasoning about compartments are similar to those about compartments in Ambient Logic; the main difference is that now boundaries are structured and not only names. Clearly, these rules do not apply to membrane logic, since membranes are not structured in compartments.

Rules for Compartments

$$(\complement\mathcal{A}\eth\neg\diamond)\ \frac{\mathcal{A}\vdash\neg\diamond}{\mathcal{M}\complement\mathcal{A}\eth\vdash\neg\diamond} \qquad (\mathcal{M}\complement\eth\neg\diamond)\ \frac{\mathcal{M}\vdash\neg\mathbf{0}}{\mathcal{M}\complement\mathcal{A}\eth\vdash\neg\diamond}$$

$$(\mathbf{0}\complement\diamond\eth)\ \frac{}{\mathbf{0}\complement\diamond\eth\dashv\vdash\diamond} \qquad (\mathcal{M}\complement\eth\neg\circ)\ \frac{}{\mathcal{M}\complement\mathcal{A}\eth\vdash\neg(\neg\diamond\circ\neg\diamond)}$$

$$(\mathcal{M}\complement\eth\vdash)\ \frac{\mathcal{A}\vdash\mathcal{B} \quad \mathcal{M}\vdash\mathcal{N}}{\mathcal{M}\complement\mathcal{A}\eth\vdash\mathcal{N}\complement\mathcal{B}\eth} \qquad (\mathcal{M}\complement\eth\wedge)\ \frac{}{\mathcal{M}\complement\mathcal{A}\eth\wedge\mathcal{M}\complement\mathcal{B}\eth\vdash\mathcal{M}\complement\mathcal{A}\wedge\mathcal{B}\eth}$$

$$(\mathcal{M}\complement\eth@)\ \frac{\mathcal{M}\complement\mathcal{A}\eth\vdash\mathcal{B}}{\mathcal{A}\vdash\mathcal{B}@\mathcal{M}} \qquad (\mathcal{M}\complement\eth\vee)\ \frac{}{\mathcal{M}\complement\mathcal{A}\vee\mathcal{B}\eth\vdash\mathcal{M}\complement\mathcal{A}\eth\vee\mathcal{M}\complement\mathcal{B}\eth}$$

$$(\neg@)\ \frac{}{\mathcal{A}@\mathcal{M}\dashv\vdash\neg(\neg(\mathcal{A})@\mathcal{M})}$$

The first two rules state that a compartment cannot be considered non-existent if the membrane is not empty or the contained system is not empty. The third rule states that an inactive membrane enclosing an empty system is logically equivalent to an empty system. The fourth rule states that a single compartment cannot be decomposed into two non-trivial systems. The rule ($\mathcal{M}\{\!|\mathcal{D}@$) shows that $\mathcal{A}@\mathcal{B}$ and $\mathcal{M}\{\!|\mathcal{A}\mathcal{D}$ are adjoints, and the rule ($\neg@$) that the compartment adjoint @ is self-dual.

The fragment about compartment is particularly simple to handle, because all rules (with assumptions) are bidirectional: ($\mathcal{M}\{\!|\mathcal{D}\vdash$) holds in both directions, and the inverses of ($\mathcal{M}\{\!|\mathcal{D}\wedge$) and ($\mathcal{M}\{\!|\mathcal{D}\vee$) are derivable.

See [11] for some corollaries about compartments.

Time and space modalities Let us now discuss the logical rules and properties about spatial and temporal modalities.

Some rules for spatial and temporal modalities in systems

$$(\Diamond\mathcal{M}\{\!|\mathcal{D}) \ \frac{}{\mathcal{M}\{\!|\Diamond\mathcal{A}\mathcal{D}\vdash\Diamond\mathcal{M}\{\!|\mathcal{A}\mathcal{D}} \qquad (\Diamond\circ) \ \frac{}{\Diamond\mathcal{A}\circ\Diamond\mathcal{B}\vdash\Diamond(\mathcal{A}\circ\mathcal{B})} \qquad (\Diamond\Diamond) \ \frac{}{\Diamond\Diamond\mathcal{A}\vdash\Diamond\Diamond\mathcal{A}}$$

$$(\Diamond\mathcal{M}\{\!|\mathcal{D}) \ \frac{}{\mathcal{M}\{\!|\Diamond\mathcal{A}\mathcal{D}\vdash\Diamond\mathcal{M}\{\!|\mathcal{A}\mathcal{D}} \qquad (\Diamond\circ) \ \frac{}{\Diamond\mathcal{A}\circ\Diamond\mathcal{B}\vdash\Diamond(\mathcal{A}\circ\mathcal{B})} \qquad (\Diamond\Diamond) \ \frac{}{\Diamond\Diamond\mathcal{A}\vdash\Diamond\Diamond\mathcal{A}}$$

The rules for these constructors are very similar to those of ambient logic [5]. The modalities $\Diamond$ and $\Diamond$ obey the rules of S4 modalities, but are not S5 modalities [9]. The last rule shows that the two modalities permute in one direction. The other direction does not hold; consider, e.g., the formula $\mathcal{A}=\langle\!|\mathfrak{o}_k\rangle\!|\{\!|\mathcal{D}$ and the system $P=\mathfrak{o}_m^{\perp}\{\!|\mathfrak{o}_m\{\!|\mathfrak{o}_n\{\!|\mathcal{D}\mathcal{D}\mathcal{D}\circ\mathfrak{o}_n^{\perp}(\mathfrak{o}_k)\{\!|\mathcal{D}$. Then, $P\vDash\Diamond\Diamond\mathcal{A}$, but $P\nvDash\Diamond\Diamond\mathcal{A}$ because neither P nor any of its subsystems will ever exhibit the action $\mathfrak{o}_k$.

On the other hand, the action modality $\langle\!|\alpha\rangle\!|\mathcal{M}$ of membranes does not satisfy the laws of S4 modality, because the relation $\xrightarrow{\alpha}$ is neither reflexive nor transitive. Nevertheless, it satisfies the laws of any Kripke modality [9].

Rules for action modality

$$(\langle\!|\alpha\rangle\!|) \ \frac{}{\langle\!|\alpha\rangle\!|\mathcal{M}\vdash\neg\,[\alpha]\,\neg\mathcal{M}} \qquad ([\alpha]\,K) \ \frac{}{[\alpha]\,(\mathcal{M}\Rightarrow\mathcal{N})\vdash[\alpha]\,\mathcal{M}\Rightarrow[\alpha]\,\mathcal{N}} \qquad ([\alpha]\vdash) \ \frac{\mathcal{M}\vdash\mathcal{N}}{[\alpha]\,\mathcal{M}\vdash[\alpha]\,\mathcal{N}}$$

Some corollaries about action modality

$$([\alpha]) \ \frac{}{[\alpha]\,\mathcal{M}\vdash\neg\langle\!|\alpha\rangle\!|\neg\mathcal{M}} \qquad (\langle\!|\alpha\rangle\!|K) \ \frac{}{\langle\!|\alpha\rangle\!|\mathcal{M}\Rightarrow\langle\!|\alpha\rangle\!|\mathcal{N}\vdash\langle\!|\alpha\rangle\!|(\mathcal{M}\Rightarrow\mathcal{N})}$$

$$(\langle\!|\alpha\rangle\!|\vdash) \ \frac{\mathcal{M}\vdash\mathcal{N}}{\langle\!|\alpha\rangle\!|\mathcal{M}\vdash\langle\!|\alpha\rangle\!|\mathcal{N}} \qquad ([\alpha]\wedge) \ \frac{}{[\alpha]\,(\mathcal{M}\wedge\mathcal{N})\dashv\vdash[\alpha]\,\mathcal{M}\wedge[\alpha]\,\mathcal{N}}$$

$$([\alpha]\,\langle\!|\alpha\rangle\!|) \ \frac{}{[\alpha]\,\mathcal{M}\vdash\langle\!|\alpha\rangle\!|\mathcal{M}} \qquad (\langle\!|\alpha\rangle\!|\vee) \ \frac{}{\langle\!|\alpha\rangle\!|(\mathcal{M}\vee\mathcal{N})\vdash\langle\!|\alpha\rangle\!|\mathcal{M}\vee\langle\!|\alpha\rangle\!|\mathcal{N}}$$

A quite expressive set of rules can be obtained by *reflecting* at the logical level the operational behaviour of systems and membranes. The next table shows some of these rules, which can be validated using the reaction of the calculus.

Logical rules for reactions

$$(\langle\mathfrak{d}\rangle)\ \frac{}{\langle\mathfrak{d}_n\rangle\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\circ\langle\mathfrak{d}_n^{\perp}(\mathcal{K})\rangle\mathcal{N}\mathbb{C}\mathcal{B}\mathbb{D}\vdash\Diamond\mathcal{N}\mathbb{C}\mathcal{K}\mathbb{C}\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\mathbb{D}\circ\mathcal{B}\mathbb{D}}$$

$$(\langle\mathfrak{d}\rangle)\ \frac{}{\langle\mathfrak{d}_n^{\perp}\rangle\mathcal{N}\mathbb{C}\langle\mathfrak{d}_n\rangle\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\circ\mathcal{B}\mathbb{D}\vdash\Diamond(\mathcal{M}|\mathcal{N}\mathbb{C}\mathcal{B}\mathbb{D}\circ\mathcal{A})}$$

$$(\langle\circledcirc\rangle)\ \frac{}{\langle\circledcirc(\mathcal{N})\rangle\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\vdash\Diamond\mathcal{M}\mathbb{C}\mathcal{N}\mathbb{C}\diamond\mathbb{D}\circ\mathcal{A}\mathbb{D}}$$

Some corollaries about reactions

$$(\langle\mathsf{mate}\rangle)\ \frac{}{\langle\mathsf{mate}_n\rangle\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\circ\langle\mathsf{mate}_n^{\perp}\rangle\mathcal{N}\mathbb{C}\mathcal{B}\mathbb{D}\vdash\Diamond\mathcal{M}|\mathcal{N}\mathbb{C}\mathcal{A}\circ\mathcal{B}\mathbb{D}}$$

$$(\langle\mathsf{bud}\rangle)\ \frac{}{\langle\mathsf{bud}_n^{\perp}(\mathcal{K})\rangle\mathcal{N}\mathbb{C}\langle\mathsf{bud}_n\rangle\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\circ\mathcal{B}\mathbb{D}\vdash\Diamond(\mathcal{K}\mathbb{C}\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\mathbb{D}\circ\mathcal{N}\mathbb{C}\mathcal{B}\mathbb{D})}$$

$$(\langle\mathsf{drip}\rangle)\ \frac{}{\langle\mathsf{drip}_n(\mathcal{N})\rangle\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D}\vdash\Diamond(\mathcal{N}\mathbb{C}\diamond\mathbb{D}\circ\mathcal{M}\mathbb{C}\mathcal{A}\mathbb{D})}$$

These rules show the connections between action modalities $\langle a\rangle$ (in the logic of membranes) and temporal modalities $\Diamond$ (in the logic of systems). These rules are very useful in verifying dynamic properties of systems and membranes.

Predicates We need to extend the notion of validity to open formulas. Let $\mathsf{FV}(\mathcal{A}) = \{x_1 \dots x_k\}$ be the set of free variables of a formula $\mathcal{A}$, and $\phi \in \mathsf{FV}(\mathcal{A}) \to \Lambda$ a substitution of names for variables; $\mathcal{A}_\phi$ denotes the formula $\mathcal{A}\{x_1 \leftarrow \phi(x_1), \dots, x_n \leftarrow \phi(x_k)\}$ obtained by applying the substitution ϕ. Then,

$$\mathbf{vld}(\mathcal{A}) \triangleq \forall\phi \in \mathsf{FV}(\mathcal{A}) \to \Lambda.\forall P \in \Pi.P \vDash \mathcal{A}_\phi$$

Using this notion of validity of formulas, the definitions of sequents and rules do not need to be changed. Then, the rules for the quantifiers are the usual ones:

Rules for the universal quantifier

$$(\forall L)\frac{\mathcal{A}\{x \leftarrow \eta\} \vdash \mathcal{B}}{\forall x.\mathcal{A} \vdash \mathcal{B}} \qquad (\forall R)\frac{\mathcal{A} \vdash \mathcal{B}}{\mathcal{A} \vdash \forall x.\mathcal{B}} \quad (x \notin \mathsf{FV}(\mathcal{A}))$$

With respect to Ambient Logic, name quantification has a slightly different meaning. In the Brane Calculus, different names are intended to denote different proteine complexes on membranes; an action and a coaction can trigger a reaction only if they are using matching complexes, i.e., names. Given this interpretation, using the quantifiers we can express properties which are schematic with respect to the names involved, that is, they do not depend on the specific complexes. For instance, $\forall x.(\langle\mathfrak{d}_x^{\perp}\rangle\mathbb{C}\langle\mathfrak{d}_x\rangle\mathbb{C}\diamond\mathbb{D}\mathbb{D} \Rightarrow \Diamond\diamond)$ means "if, for any given complexes, the system exhibits a matching exo and co-exo capabilities in the right places, then it can evolve (into the empty system)".

Name equality We can encode name equality just using logical constructors, and in particular the adjoint of compartment:

$$\eta = \mu \ \triangleq\ \langle\mathfrak{d}_\eta\rangle\mathbb{C}\mathbf{T}\mathbb{D}@\langle\mathfrak{d}_\mu\rangle$$

Proposition 2. $\forall \phi \in \mathsf{FV}(\eta, \mu) \to \Lambda . \forall P \in \Pi . P \vDash (\eta = \mu)_\phi \iff \phi(\eta) = \phi(\mu)$

As an example application, the formula

$$\forall x . \forall y . \langle\!(\mathfrak{O}_x)\!\rangle \mathbf{T} \mathbb{T} \mathbf{T} \mathbb{D} \circ \langle\!(\mathfrak{O}_y^+(\mathbf{T})\!\rangle \mathbf{T} \mathbb{T} \mathbf{T} \mathbb{D} \circ \mathbf{T} \Rightarrow \neg x = y$$

means "no pair of membranes exhibit matching action and coaction for a phagocitosis", which can be seen as a safety property (think, e.g., of a virus trying to enter a cell, and looking for the right complexes on its surface).

Substitution The next result provides a substitution principle for validity of predicates; this will allow us to replace logically equivalent formulas inside formula contexts. Let $\mathcal{B}\{-\}$ be a formula with a hole, and let $\mathcal{B}\{\mathcal{A}\}$ the formula obtained by filling the hole with $\mathcal{A}$.

Lemma 1 (Substitution). $\mathbf{vld}(\mathcal{A}' \iff \mathcal{A}'') \Rightarrow \mathbf{vld}(\mathcal{B}\{\mathcal{A}'\} \iff \mathcal{B}\{\mathcal{A}''\})$

Corollary 1 (Principle of substitution). $\mathcal{A}' \dashv\vdash \mathcal{A}'' \Rightarrow \mathcal{B}\{\mathcal{A}'\} \dashv\vdash \mathcal{B}\{\mathcal{A}''\}$

4.3 From validity of propositions to validity of predicates

We can take advantage of (name) equality to lift validity of propositions to validity of quantified formulas. As a consequence, all the rules and corollaries we have given so far for propositional validity, can be lifted to predicate validity.

To this end, we need to prove the following proposition:

Proposition 3 (Lifting propositional validity). *Let $\mathcal{A}$ be a closed valid formula. For any injective function $\psi \in \mathsf{FN}(\mathcal{A}) \to \vartheta$ mapping names to variables, the formula $(dfn(\mathcal{A}) \Rightarrow \mathcal{A})_\psi$ is valid, where $dfn(\mathcal{A}) \triangleq \bigwedge_{n,m \in \mathsf{FN}(\mathcal{A}), n \neq m} \neg(n = m)$.*

For instance, the valid proposition $[\mathfrak{O}_n]\,\mathcal{M} \Rightarrow \neg\langle\!(\mathfrak{O}_m)\!\rangle \mathcal{M}$ is mapped into the valid predicate $\neg x = y \Rightarrow ([\mathfrak{O}_x]\,\mathcal{M} \Rightarrow \neg\langle\!(\mathfrak{O}_y)\!\rangle \mathcal{M})$. Notice that without the inequalities between variables denoting different names, the result would not hold.

The proof of Proposition 3 relies on some *injective renaming* lemmata. This kind of lemmata, stating that the relevant meta-logical properties are preserved by name permutations, is quite common among calculi with names (they occur, e.g., in π-calculus, ambient calculus,...); the general technique for their proof is to proceed by induction on the syntax of formulas.

Lemma 2 (Fresh renaming preserves satisfaction).

1. *Let $\mathcal{M}$ be a closed membrane formula, σ a membrane and m, m' names such that $m' \notin \mathsf{FN}(\sigma) \cup \mathsf{FN}(\mathcal{M})$. Then, $\sigma \vDash \mathcal{M} \iff \sigma\{m \leftarrow m'\} \vDash \mathcal{M}\{m \leftarrow m'\}$.*
2. *Let $\mathcal{A}$ be a closed system formula, P a system and m, m' names such that $m' \notin \mathsf{FN}(P) \cup \mathsf{FN}(\mathcal{A})$. Then, $P \vDash \mathcal{A} \iff P\{m \leftarrow m'\} \vDash \mathcal{A}\{m \leftarrow m'\}$.*

Lemma 3 (Fresh renaming preserves validity). *Let $\mathcal{A}$ be a valid closed formula.*

1. *If m' is a name such that $m' \notin \mathsf{FN}(\mathcal{A})$, then $A\{m \leftarrow m'\}$ is closed and valid.*
2. *If $\phi \in \mathsf{FN}(\mathcal{A}) \to \Lambda$ is an injective renaming, then $\mathcal{A}_\phi$ is closed and valid.*

4.4 Example: Viral Infection

As an example of the expressivity of Brane Logic, we give the formulas describing a viral infection. We borrow the example of the Semliki Forest virus in [3].

Viral infection system

$$
\begin{array}{ll}
\textbf{virus} & \triangleq \mathfrak{O}_n.\mathfrak{O}_k(\!|\textbf{nucap}|\!) \\
\textbf{cell} & \triangleq \textbf{membrane}(\!|\textbf{cytosol}|\!) \\
\textbf{membrane} & \triangleq !\mathfrak{O}^{\perp}_n(\mathsf{mate}_m)|!\mathfrak{O}^{\perp}_w \\
\textbf{cytosol} & \triangleq \textbf{endosome} \circ Z \\
\textbf{endosome} & \triangleq !\mathsf{mate}^{\perp}_m|!\mathfrak{O}^{\perp}_k(\!|\,|\!) \\
\textbf{infected cell} & \triangleq \textbf{membrane}(\!|\textbf{nucap} \circ \textbf{cytosol}|\!)
\end{array}
$$

It is simple to show that **cell**, if placed next to **virus**, evolves into **infected cell**

$$\textbf{virus} \circ \textbf{cell} \rightarrow^{*} \textbf{infected cell}$$

The system describe in detail an infection of the Semliki Forest virus; however, it is almost impossible to abstract from the structure of the system, for instance if we are interested only in its dynamic behaviour. There are entire subsystems (e.g. Z) or parts of mebranes (e.g. $!\mathfrak{O}_w$) in **cell** that are not involved in the infection process. These are only a burden in explaining what happens in the infection process. The logic can help us to abstract from these irrelevant details: the formulas describe only what is really needed for the viral attack to take place. This kind of abstraction is very important in more complex systems or for focusing only about certain aspects of their evolution.

$$
\begin{array}{ll}
\mathit{Virus} & \triangleq \langle\!|\mathfrak{O}_n|\!\rangle\langle\!|\mathfrak{O}_k|\!\rangle\mathbf{T}(\!|\mathit{Nucap}|\!) \\
\mathit{InfectableCell} & \triangleq \exists x.\mathit{Membrane}(x)(\!|\mathit{Endosome}(x)^{\exists}|\!) \\
\mathit{Membrane}(x) & \triangleq \langle\!|\mathfrak{O}^{\perp}_n(\langle\!|\mathsf{mate}_x|\!\rangle T)|\!\rangle\mathbf{T} \\
\mathit{Endosome}(x) & \triangleq \langle\!|\mathsf{mate}^{\perp}_x|\!\rangle\mathbf{T}|\langle\!|\mathfrak{O}^{\perp}_k|\!\rangle\mathbf{T}(\!|\mathbf{T}|\!) \\
\mathit{InfectedCell} & \triangleq \mathbf{T}(\!|\mathit{Nucap}^{\exists}|\!)
\end{array}
$$

A system satisfies *Virus* if and only if it can be phagocitated by cells revealing a co-phago action with key n on their surface, and, after that, it can release its nucleocapsid if enveloped in a membrane revealing a co-exo action with key k. An infectable cell is a cell containing an endosome, such that their respective membranes have matching mate and $\mathsf{mate}^{\perp}$ actions and which exhibit the keys requested by $\mathfrak{O}$ and $\mathfrak{O}$ actions of the virus. Notice that the existential quantifier allow us to abstract from the specific key x in the membrane and the endosome: it is not important which is the specific key, only that it is the same.

Using the logical rules, we can derive that "an infectable cell can become infected if it gets close to a virus":

$$\mathit{InfectableCell} \vdash \mathit{Virus} \rhd \Diamond \mathit{InfectedCell}$$

5 A decidable sublogic

In this section we describe a simple model checker for a decidable fragment of the Brane Logic. On the basis of undecidability results for model checking of Ambient Logic [6], we expect that the statement "$P \vDash \mathcal{A}$" is undecidable. There are several reasons for this. First, replication allows to define infinitary systems and membranes. Restricting to replication-free processes and membranes does not suffice either; in fact, following [6], it should be possible to reduce the finite model problem of first order logic to model checking of replication-free systems against first order formulas extended with compartements, composition and compositionadjoint. However, it is possible to consider fragments of the logic, where model checking is decidable. In this section, we describe a model checker for replication-free systems against adjoint-free formulas. Although this logic is not very expressive, it allows to point out the differences respect to the model checker presented in [5], especially in the verification of membrane satisfaction.

5.1 Deciding satisfaction of membrane formulas

Let us consider first the problem of deciding "$\sigma \vDash \mathcal{M}$", where σ is a !-free membrane and $\mathcal{M}$ is an $\blacktriangleright$-free membrane formula. This problem can be solved without checking system formulas. As a first step, every !-free membrane can be put in a normal form, given by a finite multiset of *prime membranes*.

Normalization of a replication-free membrane

$$\xi ::= \mathbf{0} \mid a.\sigma \qquad \text{(prime membranes)}$$

$$\begin{aligned} \mathsf{Norm}(\mathbf{0}) &\triangleq [] \qquad & \mathsf{Norm}(a.\sigma) &\triangleq [a.\sigma] \\ \mathsf{Norm}(\sigma|\tau) &\triangleq [\xi_1, \ldots, \xi_k, \xi'_1, \ldots, \xi'_l], \\ & \quad \text{where } \mathsf{Norm}(\sigma) = [\xi_1, \ldots, \xi_k] \text{ and } \mathsf{Norm}(\tau) = [\xi'_1, \ldots, \xi'_l] \end{aligned}$$

Lemma 4. *If* $\mathsf{Norm}(\sigma) = [\xi_1, \ldots, \xi_k]$ *then* $\sigma \equiv \prod_{i=1\ldots k} \xi_i$.

The model checker algorithm for membranes consists of three mutually recursive functions: the model checker $\mathsf{Check} : \Sigma \times \Omega \to \mathit{Bool}$, an auxiliary checker $\mathsf{Check} : \Xi \times \Theta \to \mathit{Bool}$ for checking action formulas, and a function $\mathsf{Next} : \Sigma \times \Theta \to \mathcal{P}_f(\Xi)$. Intuitively, $\mathsf{Next}(\sigma, \alpha)$ is the (finite) set of residuals of σ after performing an action satisfying α.

Checking whether membrane σ satisfies closed formula $\mathcal{M}$

$$\begin{aligned} \mathsf{Check}(\sigma, \mathbf{T}) &\triangleq \mathbf{T} \\ \mathsf{Check}(\sigma, \neg\mathcal{M}) &\triangleq \neg\mathsf{Check}(\sigma, \mathcal{M}) \\ \mathsf{Check}(\sigma, \mathcal{M} \vee \mathcal{N}) &\triangleq \mathsf{Check}(\sigma, \mathcal{M}) \vee \mathsf{Check}(\sigma, \mathcal{N}) \\ \mathsf{Check}(\sigma, \mathbf{0}) &\triangleq \mathsf{Norm}(\sigma) = [] \\ \mathsf{Check}(\sigma, \mathcal{M}|\mathcal{N}) &\triangleq \text{let } \mathsf{Norm}(\sigma) = [\xi_1, \ldots, \xi_k] \text{ in} \\ &\qquad \exists I, J. I \cup J = \{1, \ldots, k\} \wedge I \cap J = \emptyset \wedge \\ &\qquad \mathsf{Check}(\textstyle\prod_{i \in I} \xi_i, \mathcal{M}) \wedge \mathsf{Check}(\textstyle\prod_{j \in J} \xi_j, \mathcal{N}) \\ \mathsf{Check}(\sigma, \langle\alpha\rangle\mathcal{M}) &\triangleq \exists \tau \in \mathsf{Next}(\sigma, \alpha). \mathsf{Check}(\tau, \mathcal{M}) \end{aligned}$$

$$\begin{aligned}
\mathsf{Next}(\mathbf{0},\alpha) &\triangleq \emptyset \\
\mathsf{Next}(\sigma|\tau,\alpha) &\triangleq \mathsf{Next}(\sigma,\alpha) \cup \mathsf{Next}(\tau,\alpha) \\
\mathsf{Next}(a.\sigma,\alpha) &\triangleq \text{if } \mathsf{Check}(a,\alpha) \text{ then } \{\sigma\} \text{ else } \emptyset
\end{aligned}$$

$$\begin{aligned}
\mathsf{Check}(\mathfrak{S}_n,\mathfrak{S}_m) &\triangleq n=m & \mathsf{Check}(\mathfrak{S}^\perp_n(\sigma),\mathfrak{S}^\perp_m(\mathcal{M})) &\triangleq n=m \wedge \mathsf{Check}(\sigma,\mathcal{M}) \\
\mathsf{Check}(\mathfrak{O}_n,\mathfrak{O}_m) &\triangleq n=m & \mathsf{Check}(\circledcirc_n(\sigma),\circledcirc_m(\mathcal{M})) &\triangleq n=m \wedge \mathsf{Check}(\sigma,\mathcal{M}) \\
\mathsf{Check}(\mathfrak{O}^\perp_n,\mathfrak{O}^\perp_m) &\triangleq n=m & \mathsf{Check}(\mathsf{wrap}_n(\sigma),\mathsf{wrap}_m(\mathcal{M})) &\triangleq n=m \wedge \mathsf{Check}(\sigma,\mathcal{M}) \\
& & \mathsf{Check}(a,\alpha) &\triangleq \mathbf{F} \text{ otherwise}
\end{aligned}$$

The algorithm always terminates, because each recursive call is on formulas and membranes smaller than the original ones.

Proposition 4. *For all !-free membranes σ and $\blacktriangleright$-free closed membrane formulas $\mathcal{M}$, $\sigma \vDash \mathcal{M}$ iff $\mathsf{Check}(\sigma,\mathcal{M}) = \boldsymbol{T}$.*

5.2 Deciding satisfaction of system formulas

The model checker for system formulas relies on the model checker for membranes. First we have to define a normalization function for systems into multisets of *prime systems*.

Normalization of a replication-free system

$$\begin{aligned}
\pi &::= \diamond \mid \sigma\langle\!|P|\!\rangle && \text{(prime systems)} \\
\mathsf{Norm}(\diamond) &\triangleq [] & \mathsf{Norm}(\sigma\langle\!|P|\!\rangle) &\triangleq [\sigma\langle\!|P|\!\rangle] \\
\mathsf{Norm}(P \circ Q) &\triangleq [\pi_1,\ldots,\pi_k,\pi'_1,\ldots,\pi'_l], \\
& \quad \text{where } \mathsf{Norm}(P)=[\pi_1,\ldots,\pi_k] \text{ and } \mathsf{Norm}(Q)=[\pi'_1,\ldots,\pi'_l]
\end{aligned}$$

Lemma 5. *If $\mathsf{Norm}(P) = [\pi_1,\ldots,\pi_k]$ then $P \equiv \prod_{i=1\ldots k} \pi_i$.*

As for many modal logics, we need two auxiliary functions $\mathsf{Reach}, \mathsf{SubLoc} : \Pi \to \mathcal{P}_f(\Pi)$ for checking the two modalities. Their specification is the following:

$$\begin{aligned}
Q \in \mathsf{Reach}(P) &\Rightarrow P \blacktriangleright^* Q & \forall P'.P \blacktriangleright^* P' &\Rightarrow \exists Q \in \mathsf{Reach}(P).P' \equiv Q \\
Q \in \mathsf{SubLoc}(P) &\Rightarrow P \downarrow^* Q & \forall P'.P \downarrow^* P' &\Rightarrow \exists Q \in \mathsf{SubLoc}(P).P' \equiv Q
\end{aligned}$$

Due to lack of space, we omit their (easy) definitions.

Checking whether system P satisfies closed formula $\mathcal{A}$

$$\begin{aligned}
\mathsf{Check}(P,\mathbf{T}) &\triangleq \mathbf{T} \\
\mathsf{Check}(P,\neg\mathcal{A}) &\triangleq \neg\mathsf{Check}(P,\mathcal{A}) \\
\mathsf{Check}(P,\mathcal{A}\vee\mathcal{B}) &\triangleq \mathsf{Check}(P,\mathcal{A}) \vee \mathsf{Check}(P,\mathcal{B}) \\
\mathsf{Check}(P,\mathbf{0}) &\triangleq \mathsf{Norm}(P) = []
\end{aligned}$$

$$
\begin{aligned}
\mathsf{Check}(P, \mathcal{A}|\mathcal{B}) \triangleq & \text{let } \mathsf{Norm}(P) = [\pi_1, \ldots, \pi_k] \text{ in} \\
& \exists I, J. I \cup J = \{1, \ldots, k\} \wedge I \cap J = \emptyset \wedge \\
& \mathsf{Check}(\textstyle\prod_{i \in I} \pi_i, \mathcal{A}) \wedge \mathsf{Check}(\textstyle\prod_{j \in J} \pi_j, \mathcal{B}) \\
\mathsf{Check}(P, \mathcal{M}\mathfrak{C}\mathcal{A}\mathfrak{D}) \triangleq & \exists \sigma, Q. \mathsf{Norm}(P) = [\sigma \mathfrak{C} Q \mathfrak{D}] \wedge \mathsf{Check}(\sigma, \mathcal{M}) \wedge \mathsf{Check}(Q, \mathcal{A}) \\
\mathsf{Check}(P, \forall x. \mathcal{A}) \triangleq & \text{let } m \notin \mathsf{FN}(P) \cup \mathsf{FN}(\mathcal{A}) \text{ in} \\
& \forall n \in \mathsf{FN}(P) \cup \mathsf{FN}(\mathcal{A}) \cup \{m\}. \mathsf{Check}(P, \mathcal{A}\{x \leftarrow m\}) \\
\mathsf{Check}(P, \Diamond \mathcal{A}) \triangleq & \exists Q \in \mathsf{Reach}(P). \mathsf{Check}(Q, \mathcal{A}) \\
\mathsf{Check}(P, \Diamond\!\!\!\!- \mathcal{A}) \triangleq & \exists Q \in \mathsf{SubLoc}(P). \mathsf{Check}(Q, \mathcal{A})
\end{aligned}
$$

Also this algorithm always terminates, because each recursive call is on formulas and processes smaller than the original ones. Notice that in the case of compartment, we execute the model checker over membranes defined above.

Proposition 5. *For all !-free systems P and $(\triangleright \blacktriangleright @)$-free closed system formulas $\mathcal{A}$, $P \vDash \mathcal{A}$ iff $\mathsf{Check}(P, \mathcal{A}) = \boldsymbol{T}$.*

6 Conclusions

In this paper we have introduced a modal logic for describing spatial and temporal properties of biological systems represented as nested membranes, with particular attention to the computational activity which takes place *on* membranes. The logic is quite expressive, since it can describe in a easy but formal way a large range of biological situations at the abstraction level of membrane machines. For a decidable sublogic, we have given a model-checking algorithm, which is a useful tool for automatic verification of properties (e.g., vulnerabilities) of biological systems.

The work presented in this paper is intended to be the basis for further developments, in many directions. First, we can consider logics for more expressive brane calculi, e.g. with communication cross/on-membranes and protein complexes logic formulas. Suitable corresponding logical constructors can be added to the logic of actions. Also, the logic can be easily adapted to other variants of the Brane Calculus, such as the Projective Brane Calculus [7] (e.g., a system formula like $\langle \mathcal{M}; \mathcal{N} \rangle \mathfrak{C} \mathcal{A} \mathfrak{D}$ would carry a formula for each face of the membrane).

Another interesting aspect to investigate is the notion of logical equivalence induced by the logic. This should be similar to the equivalences induced by Hennessy-Milner logic extended with spatial connectives (for membranes) and of Ambient Logic (for systems). We think that the methodologies and results developed in [15] can be extended to our logic.

Moreover, it would be interesting to extend the decidability result to a larger class of formulas. We plan to extend the model checker algorithm to formulas without quantifiers but with the guarantees operators (i.e., the adjoints of compositions), along the lines of [6]. On a different direction, it is interesting to

consider also *epistemic logics* [10], where the role of the guarantee operator is played by an epistemic operator, while maintaining decidability.

Acknowledgments The authors wish to thank Luca Cardelli for useful discussions and for kindly providing the fancy font of the actions of Brane Calculus.

References

1. B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J. D. Watson. *Molecular biology of the cell*. Garland, second edition, 1989.
2. L. Caires. Behavioral and spatial observations in a logic for the pi-calculus. In I. Walukiewicz, editor, *FoSSaCS*, volume 2987 of *Lecture Notes in Computer Science*, pages 72–89. Springer, 2004.
3. L. Cardelli. Brane calculi. In V. Danos and V. Schachter, editors, *CMSB*, volume 3082 of *Lecture Notes in Computer Science*, pages 257–278. Springer, 2004.
4. L. Cardelli. Abstract machines of systems biology. *T. Comp. Sys. Biology*, 3737:145–168, 2005.
5. L. Cardelli and A. D. Gordon. Anytime, anywhere: Modal logics for mobile ambients. In *Proc. POPL*, pages 365–377, 2000.
6. W. Charatonik, S. Dal-Zilio, A. D. Gordon, S. Mukhopadhyay, and J.-M. Talbot. Model checking mobile ambients. *Theor. Comput. Sci.*, 308(1-3):277–331, 2003.
7. V. Danos and S. Pradalier. Projective brane calculus. In V. Danos and V. Schächter, editors, *CMSB*, volume 3082 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 2004.
8. M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985.
9. G. E. Hughes and M. J. Cresswell. *A companion to Modal Logic*. Methuen, London, 1984.
10. R. Mardare and C. Priami. A decidable extension of hennessy-milner logic with spatial operators. Technical Report DIT-06-009, Dipartimento di Informatica e Telecomunicazioni, University of Trento, 2006.
11. M. Miculan and G. Bacci. Modal logics for brane calculus. Technical Report UDMI/08/2006/RR, Dept. of Mathematics and Computer Science, Univ. of Udine, 2006. `http://www.dimi.uniud.it/miculan/Papers/UDMI082006.pdf`.
12. R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.
13. A. Regev, W. Silverman, and E. Y. Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. In *Pacific Symposium on Biocomputing*, pages 459–470, 2001.
14. J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In *LICS*, pages 55–74. IEEE Computer Society, 2002.
15. D. Sangiorgi. Extensionality and intensionality of the ambient logics. In *Proc. POPL*, pages 4–13, 2001.