CMSB 2006
Trento, October 18-19, 2006

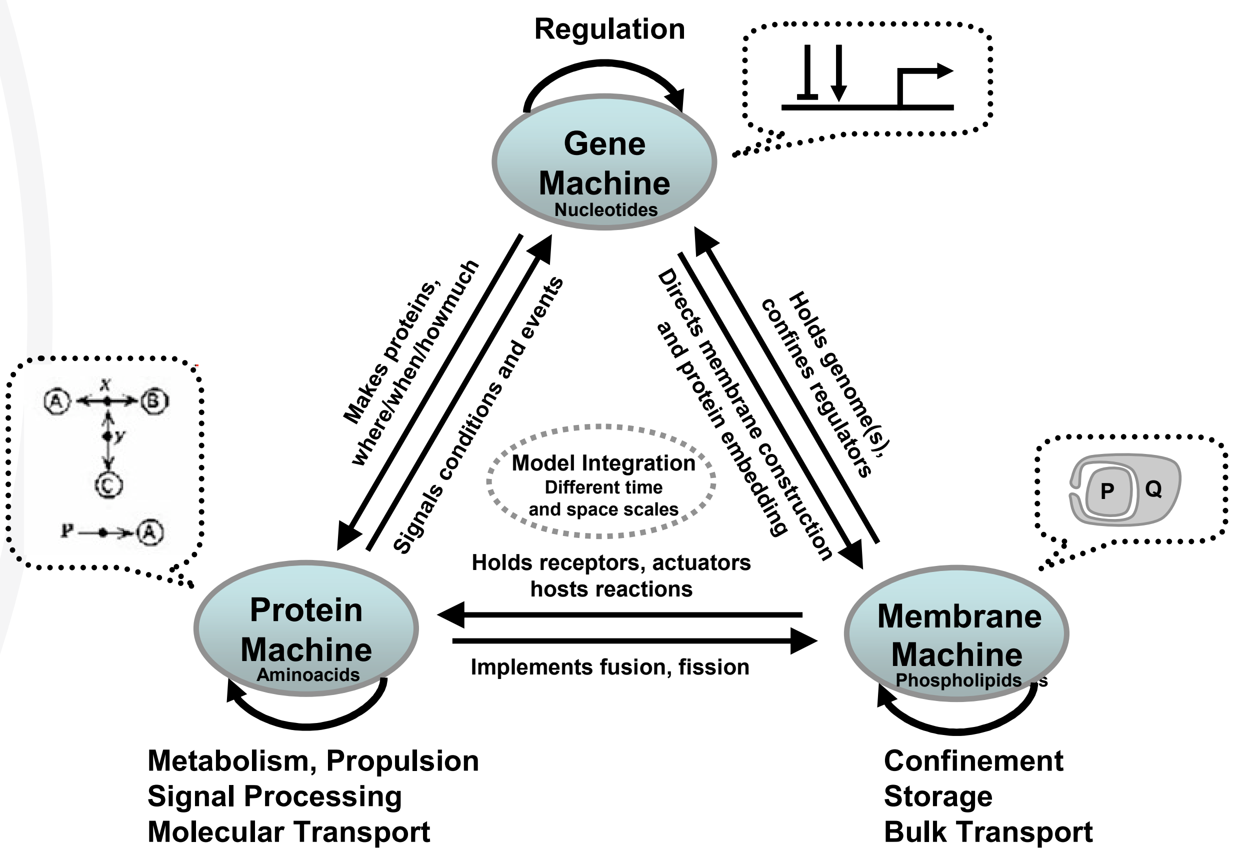
# Modal Logics for Brane Calculus

Marino Miculan
(joint work with G. Bacci)
University of Udine

# Introduction and Motivations

# Abstract Machines of Systems Biology

- Cardelli [2005] has proposed three levels of (highly interacting) abstract machines
  - **Protein machine**
  - **Gene machine**
  - **Membrane machine**




- **Strategic approach:** formalize and study each of these, and their interaction, as discrete reactive systems using tools and techniques from (Theoretical) Computer Science

# Abstract Models for Systems Biology

- **Abstract models** have been proposed for each machine (various calculi, statecharts, Petri nets…)
- These models can be used for many aims, such as:
  - formalizing biological systems (at various levels)
  - implementing tools for simulating behavior of systems
  - help biologists to understand what is really relevant
- But in Computer Science, also **logics** have been used for a while…

# Formal Methods in Comp.Sci. vs Sys.Bio.

In CS, the object system to model is man-made; in SysBio this is generally not true (for the moment)

- Ultimately we do not know how the "real thing" works
- If the model does not fit the system, we cannot ask the Designer to change His choices
- We can only *test* the model against the real world, and refine it if something goes wrong (cf. physics)

# Models as (Scientific) Theories

- Models of SysBio have to be validate **experimentally:** they hold until they are falsified by an experiment

1. formalize a system in th

How to express these properties?

2. choose some property which holds for the formal version

How to check this?

3. try an experiment to verify if the prop ds also in the real world (*predictive biology*)

4. if holds, go to 2; else go to 1 (or 0)

# Logics for Systems Biology?

MISSING

Logics allow to express formally the properties of biological systems, usually written in natural language. Some applications:

- **System specification and verification** (possibly automatic): “check whether a given system *P* satisfies a given property *A*”
- **System synthesis:** “find a system which satisfies a given property *A*” (*synthetic biology*)
- **System characterization**: “find the formula which characterizes the behaviour of a given system *P*”
- **Model validation**: predict a property which should hold in a system and mount an experiment to verify it (*predictive biology*)

# In this talk: Brane Logic

- **Brane Logic**: a logic for expressing membrane-level properties of systems described in Brane Calculus
- Plan of the rest of the talk:
  - Short recall of Brane Calculus
  - Short intro to Brane Logic
  - Examples and conclusions

# Brane Calculus

- Introduced by Cardelli (2004) as an abstract model for the membrane machine

- Similar to Ambient Calculus (due to the hierarchical structure), but computations take place *on* the membranes, not inside

- Actions are those observed at the membrane level

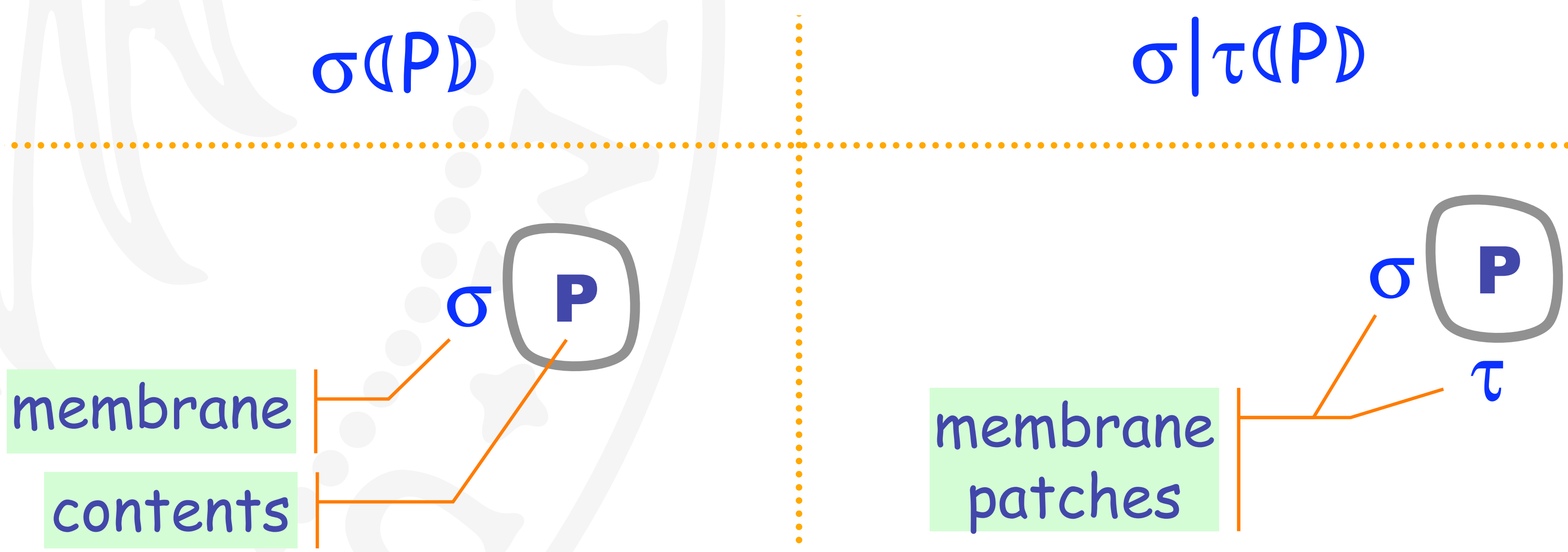
  - membrane structure interactions

  - intra- and inter-membrane communications (not considered here)

# Basic Brane calculus: Syntax

$$\begin{array}{lrl}
\text{Systems } \Pi : & P, Q ::= & \diamond \mid \sigma(\!|P|\!) \mid P \circ Q \mid !P \\
\text{Membranes } \Sigma : & \sigma, \tau ::= & \mathbf{0} \mid \sigma|\tau \mid a.\sigma \mid !\sigma \\
\text{Actions } \Xi : & a, b ::= & \circlearrowright_n \mid \circlearrowright^{\perp}_n(\sigma) \mid \circlearrowleft_n \mid \circlearrowleft^{\perp}_n \mid \circledcirc(\sigma)
\end{array}$$

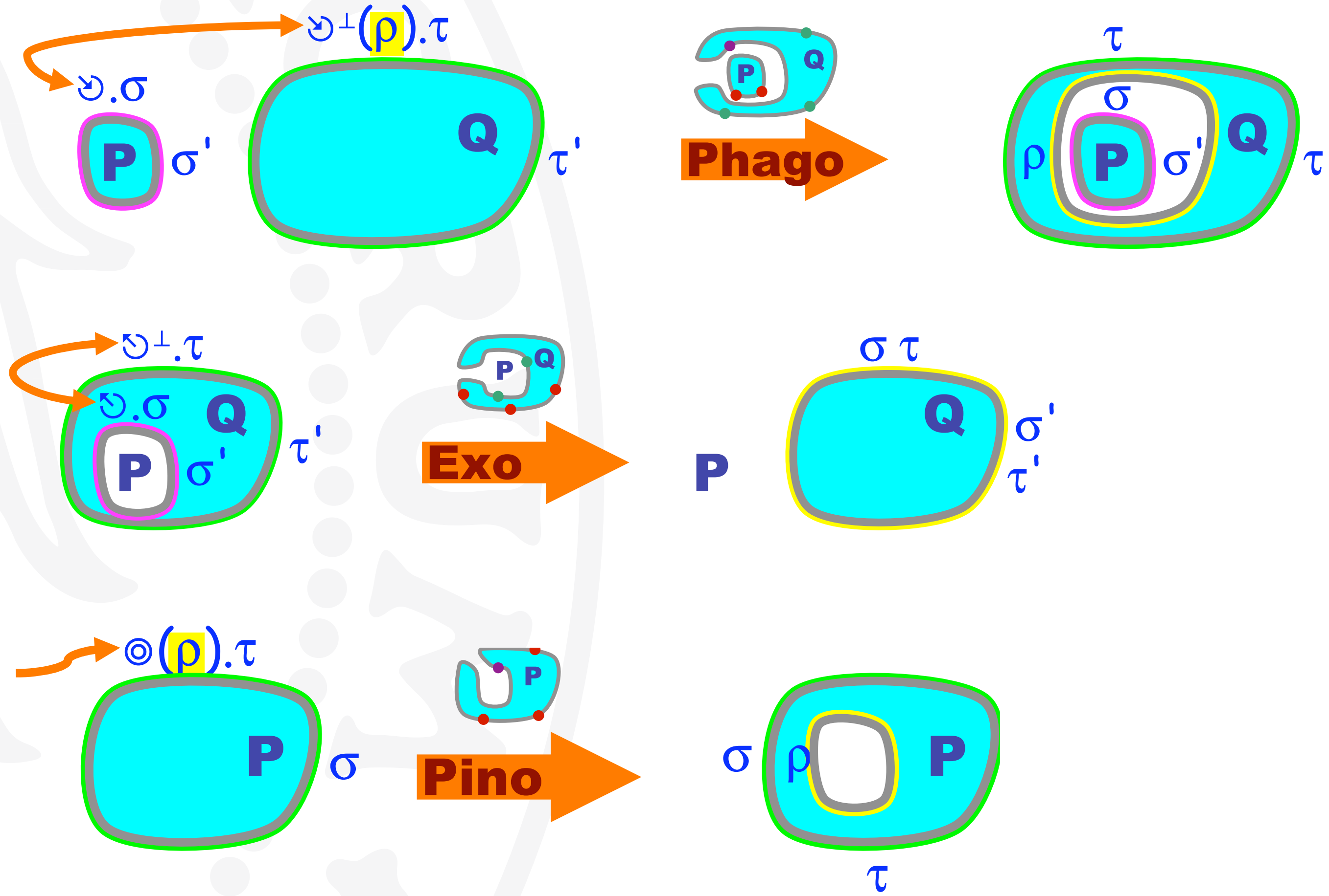
Ballons from [Cardelli 04]




- Fluidity of solutions and membranes is rendered by the usual monoidal laws of parallel compositions

# Brane Calculus: PEP Semantics

Ballons from [Cardelli 04]




**Phago** $\circlearrowleft_n.\sigma|\sigma'(\!|P|\!) \circ \circlearrowleft^{\perp}_n(\rho).\tau|\tau'(\!|Q|\!) \rightarrow \tau|\tau'(\!|\rho(\!|\sigma|\sigma'(\!|P|\!)|\!)\circ Q|\!)$

**Exo** $\circlearrowright^{\perp}_n.\tau|\tau'(\!|\circlearrowright_n.\sigma|\sigma'(\!|P|\!)\circ Q|\!) \rightarrow P \circ \sigma|\sigma'|\tau|\tau'(\!|Q|\!)$

**Pino** $\circledcirc(\rho).\sigma|\sigma'(\!|P|\!) \rightarrow \sigma|\sigma'(\!|\rho(\!|\diamond|\!)\circ P|\!)$

# Brane Logic:
# A logic for Membrane-level properties

# Design principle: “capture what we are talking of”

- The logic should be able to express properties of the membrane machine, such as th … in normal biology books:

Relative Position

Surface information

  - “If a macrophage is exposed to target cells [...] coated with antibody, it ingests the coated cells.”

State change

  - “The Rous sarcoma virus [... sform a cell into a cancer cell.”

Movement

  - “Eventually, the virus escapes from the endosome”

Time

Space

om Alberts et al., Molecular biology of the c … 9)

(Instead, *system equivalence* does not appear to be a central notion…)

# A Bi-Spatial-Temporal Modal Logic

- There are *two* interacting logics: one for membranes and one for systems
- **Spatial logic for systems** deals with compartments, like Ambient Logic - but with some differences

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

Formulas in place of names

# Logic for membranes

- Membranes are much like CCS: their logic is a kind of Hennessy-Milner (i.e. dynamic) logic with connectives for composition but not for compartment

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

- **Problem**: Hennessy-Milner logics need a labeled transition system. What is $\alpha$, the observable action?

# Which observations?

- In Hennessy-Milner logic, modalities are indexed the actions of the underlying calculus (CCS); the LTS is

$$a.\sigma \xrightarrow{a} \sigma$$

- In Brane calculus, actions may contain membranes

$$\circlearrowright(\sigma).\tau \xrightarrow{\circlearrowright(\sigma)} \tau$$

- We would observe **membranes** themselves in the formulas

- Not good: too fine-grained and intensional

# Solution: a Logic of Actions

- What we observe are **properties** of actions, not actions themselves

- **action formulas** α are the label of the membrane LTS

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

- we need to introduce a **logic of actions:**

$$\begin{array}{rll} \alpha, \beta ::= & \circlearrowleft_\eta \mid \circlearrowleft_\eta^\perp(\mathcal{M}) & (\text{phago, co-phago}) \\ & \circlearrowleft_\eta \mid \circlearrowleft_\eta^\perp & (\text{exo, co-exo}) \\ & \circledcirc(\mathcal{M}) & (\text{pino}) \end{array}$$

Membrane formulas here, not membranes!

# Satisfaction

- Satisfaction relations for the three logics are then defined as usual for spatial/temporal/HM logics. Some clauses:

$$P \models \mathcal{M}(\!|\mathcal{A}|\!) \triangleq \exists P' : \Pi, \sigma : \Sigma . P \equiv \sigma(\!|P'|\!) \wedge P' \models \mathcal{A} \wedge \sigma \models \mathcal{M}$$

$$P \models \mathcal{A}@\mathcal{M} \triangleq \forall \sigma : \Sigma . \sigma \models \mathcal{M} \Rightarrow \sigma(\!|P|\!) \models \mathcal{A}$$

$$P \models \mathcal{A} \triangleright \mathcal{B} \triangleq \forall P' : \Pi . P' \models \mathcal{A} \Rightarrow P \circ P' \models \mathcal{B}$$

$$\sigma \models \langle\alpha\rangle\mathcal{M} \triangleq \exists \sigma' : \Sigma . \sigma \xrightarrow{\alpha} \sigma' \wedge \sigma' \models \mathcal{M}$$

$$a \models \circlearrowleft_n \triangleq a = \circlearrowleft_n$$

$$a \models \circlearrowleft_n^{\perp}(\mathcal{M}) \triangleq \exists \sigma : \Sigma . a = \circlearrowleft_n^{\perp}(\sigma) \wedge \sigma \models \mathcal{M}$$

# Deciding Satisfaction

- **Proposition**: The satisfaction problem ("$P \models A$ ?") is undecidable.

  Proof similar to that of Ambient Logic (reduction to PSP)

- **Proposition**: The fragment without adjoints, against the calculus without replication, is decidable. (Model checkers for the three logics are given in the paper.)

- Conjecture: the result can be extended to finite processes against formulas with adjoints but without quantifiers (along DalZilio, Charatonik et al.)

# Proof System

- A (sound) proof system for deriving valid sequents (i.e, universally valid properties) has been given
- There are rules (induced by reduction semantics) explaining the interplay between the different logics

$$(\langle\!\langle \circlearrowright \rangle\!\rangle)\ \frac{}{\langle\!\langle \circlearrowright_n \rangle\!\rangle \mathcal{M} ⦇\mathcal{A}⦈ \circ \langle\!\langle \circlearrowright_n^{\perp}(\mathcal{K}) \rangle\!\rangle \mathcal{N} ⦇\mathcal{B}⦈ \vdash \Diamond \mathcal{N} ⦇\mathcal{K} ⦇\mathcal{M} ⦇\mathcal{A}⦈⦈ \circ \mathcal{B}⦈}$$

$$(\langle\!\langle \circlearrowleft \rangle\!\rangle)\ \frac{}{\langle\!\langle \circlearrowleft_n^{\perp} \rangle\!\rangle \mathcal{N} ⦇\langle\!\langle \circlearrowleft_n \rangle\!\rangle \mathcal{M} ⦇\mathcal{A}⦈ \circ \mathcal{B}⦈ \vdash \Diamond(\mathcal{M} | \mathcal{N} ⦇\mathcal{B}⦈ \circ \mathcal{A})}$$

$$(\langle\!\langle \circledcirc \rangle\!\rangle)\ \frac{}{\langle\!\langle \circledcirc(\mathcal{N}) \rangle\!\rangle \mathcal{M} ⦇\mathcal{A}⦈ \vdash \Diamond \mathcal{M} ⦇\mathcal{N} ⦇\diamond⦈ \circ \mathcal{A}⦈}$$

# Example: Semliki Forest Viral Infection

- Formalized in Brane Calculus [Cardelli 2004]

$$\begin{aligned}
&\textbf{virus} && \triangleq \circlearrowright_n . \circlearrowleft_k (\!|\textbf{nucap}|\!) \\
&\textbf{cell} && \triangleq \textbf{membrane} (\!|\textbf{cytosol}|\!) \\
&\textbf{membrane} && \triangleq !\circlearrowright_n^\perp(\mathsf{mate}_m) | !\circlearrowleft_w^\perp \\
&\textbf{cytosol} && \triangleq \textbf{endosome} \circ Z \\
&\textbf{endosome} && \triangleq !\mathsf{mate}_m^\perp | !\circlearrowleft_k^\perp (\!| |\!) \\
&\textbf{infected cell} && \triangleq \textbf{membrane} (\!|\textbf{n} \ldots \textbf{ol}|\!)
\end{aligned}$$

Not involved in infection

Must be matching

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

# Example (continued)

- The infection, specified in Brane Logic

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

- Only the strictly necessary parts have to be specified
- Quantifiers take care of parametric names
- We can formally derive the following sequent:

$$InfectableCell \vdash Virus \triangleright \Diamond InfectedCell$$

# Conclusions

- Introduced **Brane Logic**, a bi-spatial temporal modal logic for reasoning about Brane Calculus
- Proof system given; can be used for deriving general properties of membrane systems
- Model checker given, for a decidable fragment
- **Future work**:
  - Extend the logic with connectives for communications (bind&release)
  - Model checker for larger subset of the logic
  - Implementation: e.g. extending Delzanno work about LTL in Maude
  - Experiments…

Thanks.