Using a Timed Concurrent Constraint Process Calculus for Modeling and Verifying Biological Systems

Julián Gutiérrez, Jorge Pérez, Camilo Rueda [1] and Frank Valencia [2]

[1] Universidad Javeriana-Cali, Colombia

[2] CNRS and LIX, École Polytechnique, France

ICALP 2006

Working idea

Biological Systems modeling often requires coping with partial information

Having partial information constructs built into the modeling framework allows to

- greatly simplify system models
- observe useful approximations of system behavior
- coherently represent model refinements as experimental data become available

Partial knowledge: data and behavior

No precise knowledge of

- state: the exact value of a parameter
- behavior: the temporal occurrence of interactions

Partial information and CC calculi

In temporal CC calculi,

- precise and incomplete experimental data can be uniformly represented by constraints
- particular constraint systems can be tailored to types of data suitable to different biological domains
- non deterministic constructs can model different kinds of temporal uncertainties.

NTCC in biological modeling

We believe that NTCC has a place in biological modeling. We would like to assess its possibilities/limitations of in this domain

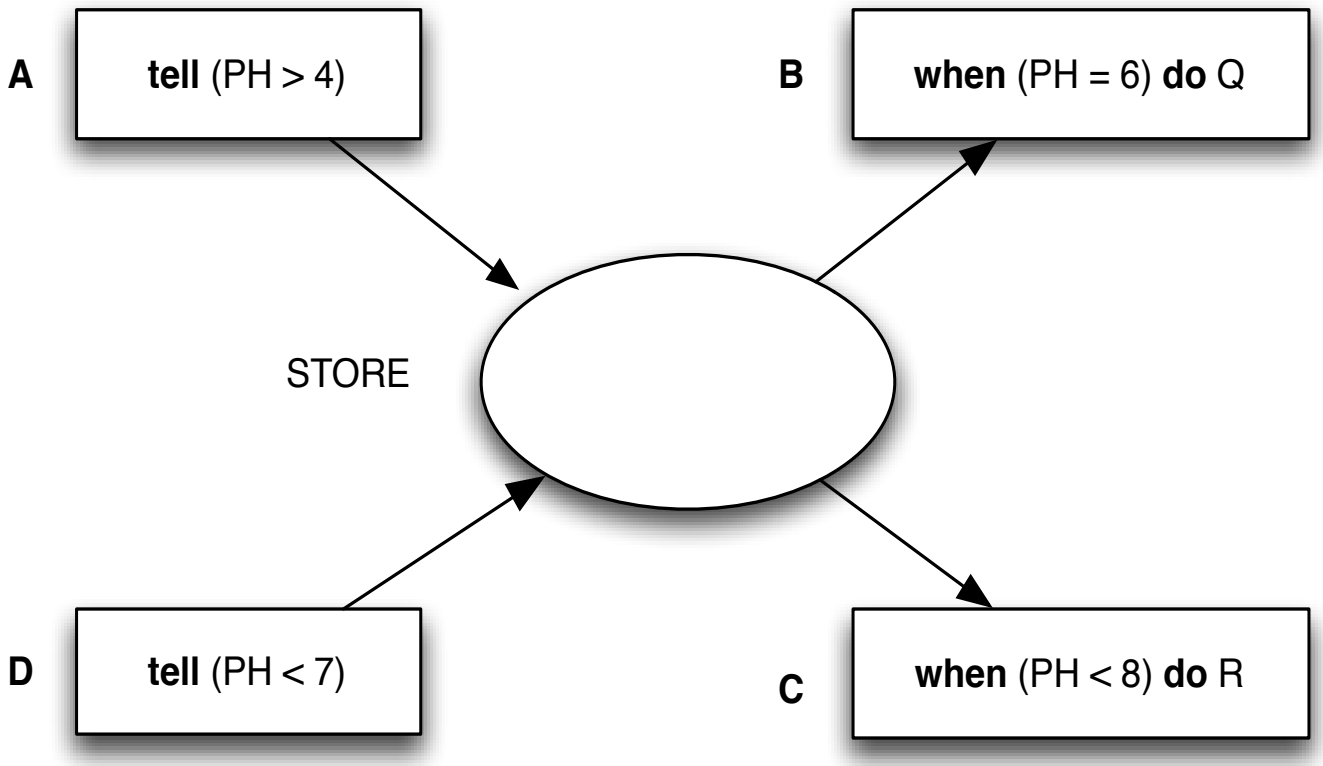
The agenda

- Introduction to the CCP model
- NTCC constructs and their semantics
- Verification of model properties
- Examples of biological models in NTCC
- Conclusions and future work

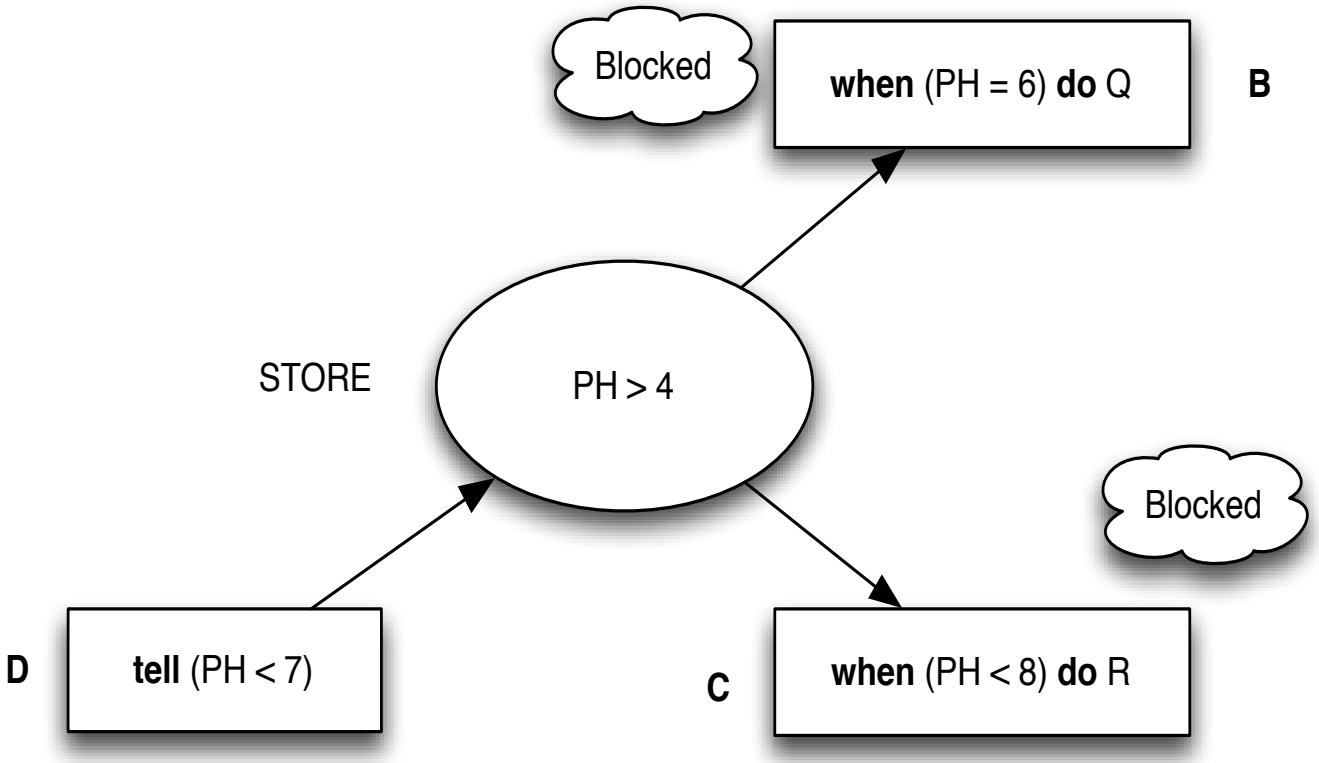
The CCP Model

- There is a global store
- Processes,
  - compute partial information and record it into the store
  - synchronize by testing (partial) data on shared variables

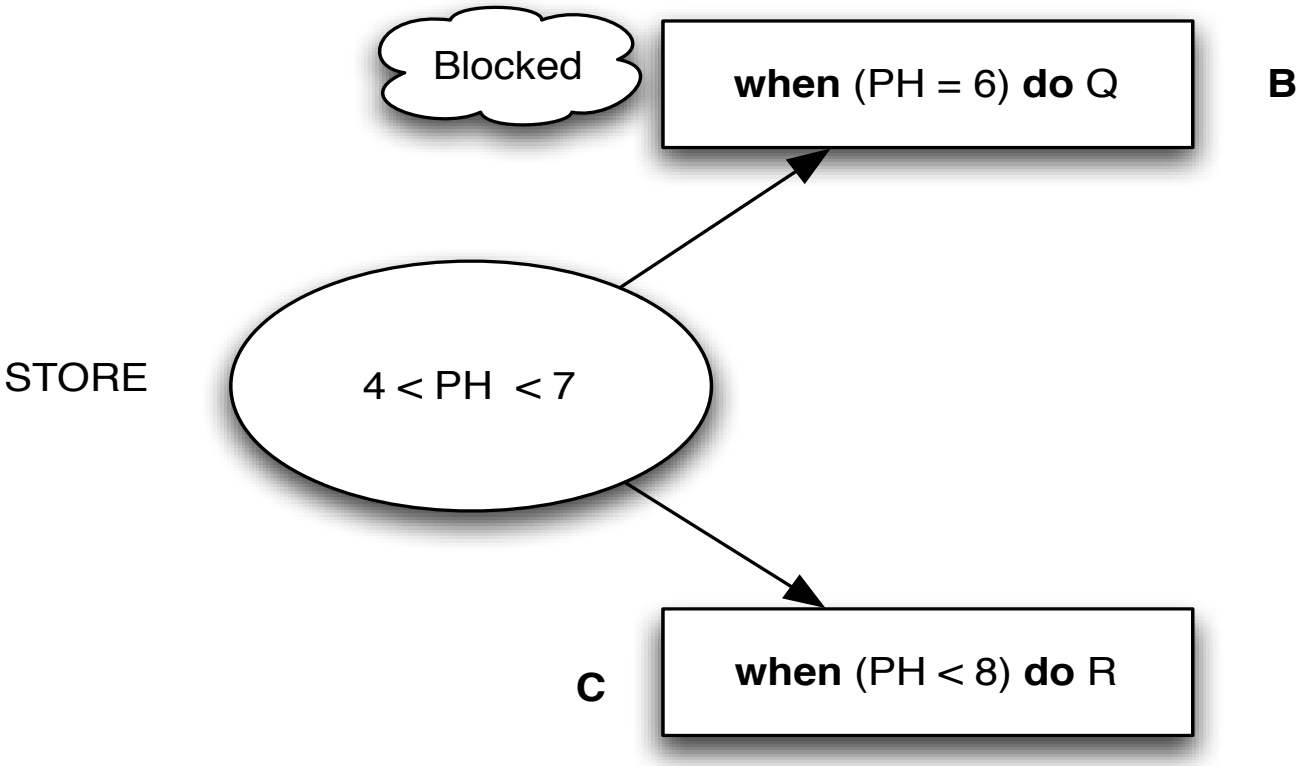
Computing with partial information in CCP



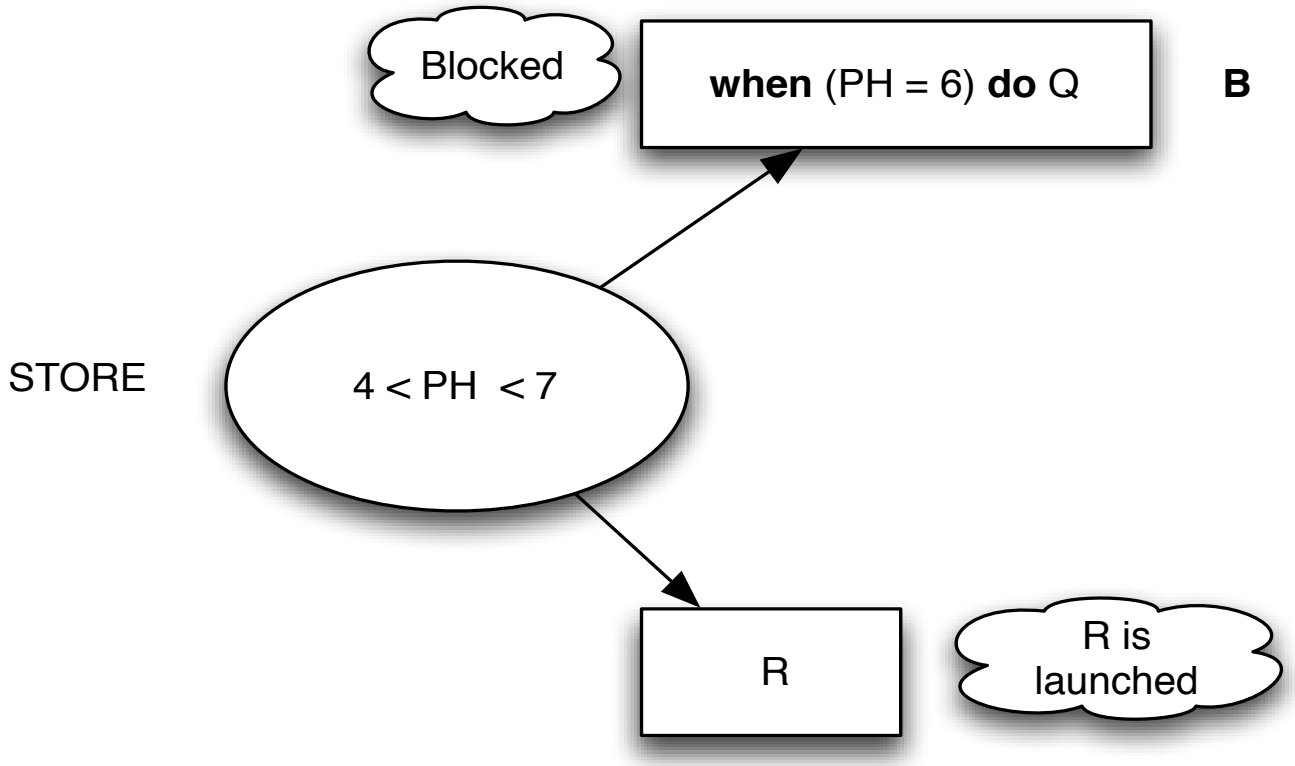
Computing with partial information in CCP



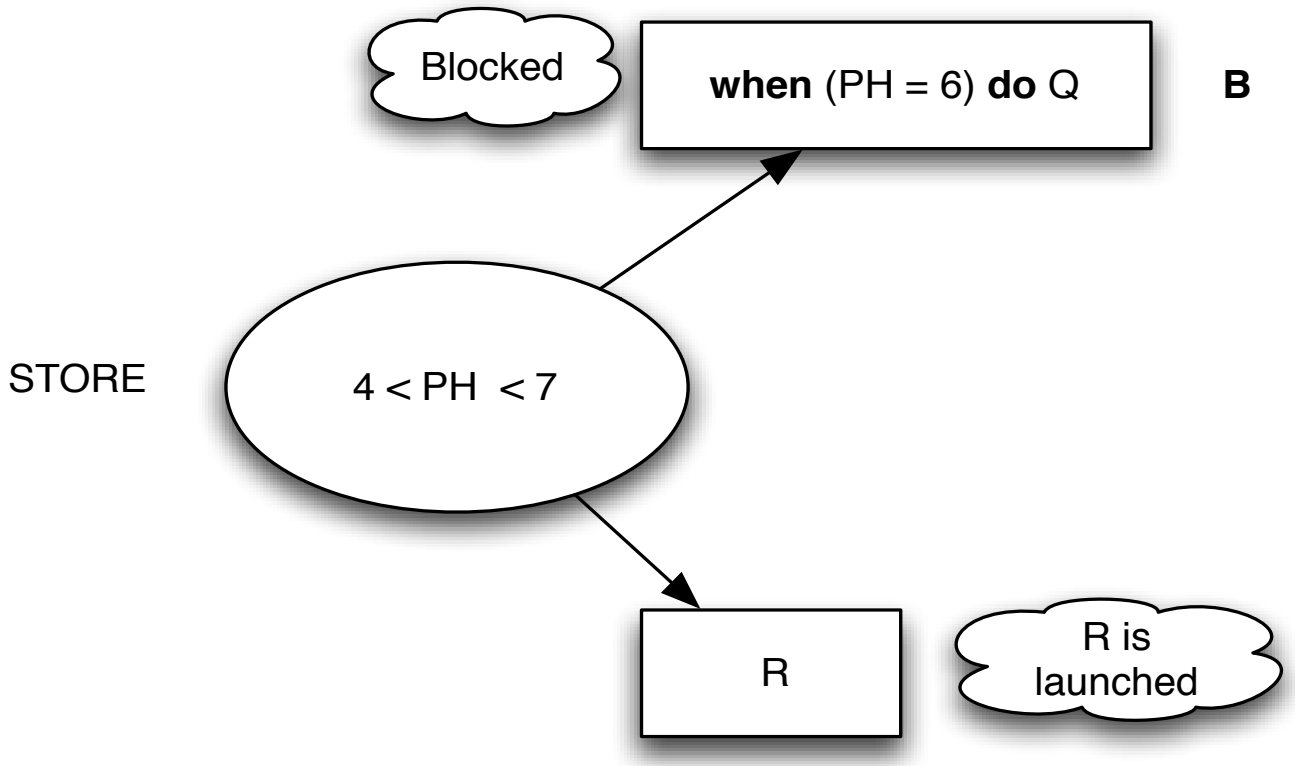
Computing with partial information in CCP



Computing with partial information in CCP



CCP basic Intuitions



- Partial Information (e.g. PH is *some unknown value* > 4)
- Concurrent execution of processes
- Synchronization via Blocking-Ask (e.g. when construct above)

Representing Partial Information

Definition. A constraint system consists of a signature Σ and first-order theory Δ over Σ

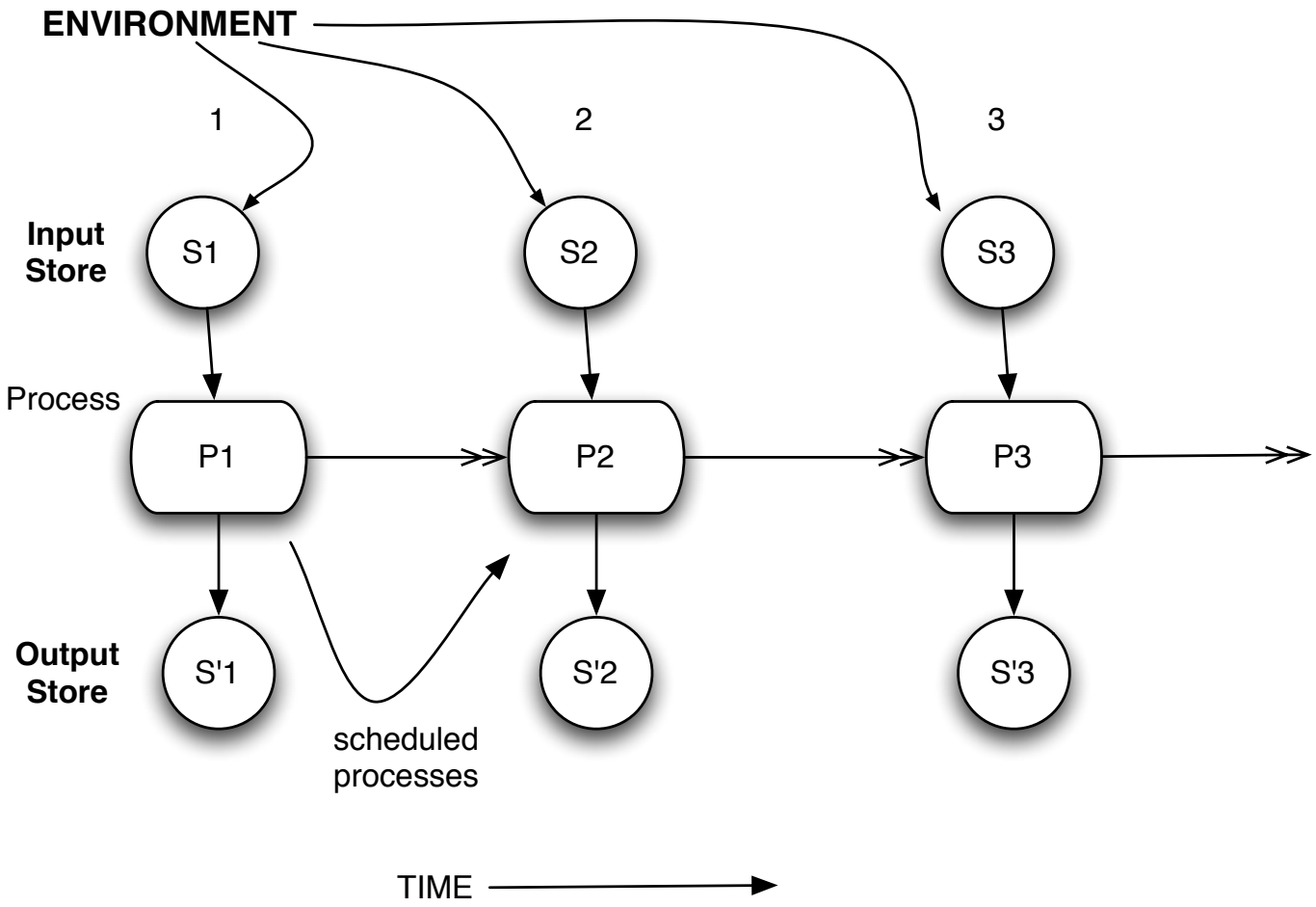
- Constraints: Formulae over Σ
- A decidable entailment relation between constraints: $\vdash_\Delta$
- A set $\mathcal{C}$ of constraints under consideration.

The NTCC calculus

Aimed at modeling reactive computation

- Parameterized on a constraint system.
- Time is divided into discrete units
- At each time unit,
  - The environment supplies some information into a store
  - Information is represented by constraints.
  - concurrent agents interact to add new constraints to the store
  - Processes can be scheduled to run in future time units

NTCC models reactive computation



NTCC Constructs

Syntax

$$tell(c)$$

Example

- Partial information : Ground rules

$$tell(l < pH_{in} < u)$$

- State definition:

$$tell(pH_{in} = f(pH_{old}, k))$$

NTCC Constructs

Syntax

$tell(c)$ $\qquad$ $when\ c\ do\ P$ $\leftarrow$

Example

Preconditions for adding information:

> *"when the level of pH reaches a threshold the range of valid values for pH_{in} should decrease"*

$$when\ pH_{in} > l * 2\ do\ tell(u' \leq u - k_1)$$

NTCC Constructs

Syntax

$tell(c)$ $\qquad$ $when\ c\ do\ P$

$\sum_{i \in I} when\ c_i\ do\ P_i \quad \leftarrow$

Example

" Non-deterministically choose some P_j $(j \in I)$ whose guard c_j is entailed by the store"

$$P = \begin{array}{l} when\ \mathrm{ATP} > 0\ do\ tell(\mathrm{releaseEnzyme} = 1) \\ + \\ when\ \mathrm{ATP} > 0\ \wedge\ \mathrm{Gradient} = 1\ do\ tell(\mathrm{emitSignal} = 1) \end{array}$$

NTCC Constructs

Syntax

$tell(c)$ $\qquad$ $when\ c\ do\ P$

$\sum_{i \in I} when\ c_i\ do\ P_i$ $\qquad$ $P \parallel Q$ $\qquad \leftarrow$

Example

Communication: two processes operating concurrently

$$Q = \begin{array}{l} when\ \text{releaseEnzyme} = 1\ do\ tell(\text{promoteReaction} = 1) \\ \quad + \\ when\ \text{emitSignal} = 1\ do\ tell(\text{promoteReaction} = 0) \end{array}$$

$P \parallel Q \longrightarrow tell(\text{promoteReaction} = 1)$ $\qquad$ (if $ATP > 0$ entailed)

NTCC Constructs

Syntax

$tell(c)$ $\quad$ $when\ c\ do\ P$

$\sum_{i \in I} when\ c_i\ do\ P_i$ $\quad$ $P \parallel Q$

$local\ x\ in\ P$ $\quad \leftarrow$

Example

Let $C_i(x_1, ..., x_n)$ be a set of complex processes (e.g. cells).
Process $\quad C_i{}^* = local\ x_p, x_{p+1}, \ldots, x_q\ in\ C_i$
narrows the context of interaction of C_i processes,
$T^* = C_1{}^* \parallel \ldots \parallel C_m{}^*$
to some set of relevant features.

NTCC Constructs

Syntax

$tell(c)$	$when\ c\ do\ P$	$\sum_{i \in I} when\ c_i\ do\ P_i$
$P \parallel Q$	$local\ x\ in\ P$	
$next\ Q$	$\leftarrow$	

Example

Defers process execution to the next time unit.

$$when\ (Na_l > Na_{IDEAL})\ do \\ next\ tell(Na_l = Na'_l - 3)$$

NTCC Constructs

Syntax

$tell(c)$	$when\ c\ do\ P$	$\sum_{i \in I} when\ c_i\ do\ P_i$	
$P \parallel Q$	$local\ x\ in\ P$		
$next\ Q$		$unless\ c\ next\ Q$	$\leftarrow$

Example

Reasoning on absence of information.

$$unless\ Na_O = Na_I\ next\ PassiveNa$$

NTCC Constructs

Syntax

$tell(c)$	$when\ c\ do\ P$	$\sum_{i \in I} when\ c_i\ do\ P_i$
$P \parallel Q$	$local\ x\ in\ P$	
$next\ Q$		$unless\ c\ next\ Q$
$\star\ Q$	$\leftarrow$	

Example

Partial information on temporal occurrence.

$$when\ M_c\ do \quad \star\ tell(Na_O = Na_I)$$

Derived construct: $\star_{[m,n]}\ Q$

Useful for describing biological processes that interact at *unknown* relative speeds

NTCC Constructs

Syntax

$tell(c)$	$when\ c\ do\ P$	$\sum_{i \in I} when\ c_i\ do\ P_i$
$P \parallel Q$	$local\ x\ in\ P$	
$next\ Q$		$unless\ c\ next\ Q$
$\star\ Q$		$!\ Q$ $\leftarrow$

Example

Persistent information.

$$!\ tell(ATP > 0)$$

Derived construct: $!_{[m,n]}\ Q$

A (very) simple bio model in NTCC

$$
\begin{array}{lcl}
\mathit{Start} & = & \mathit{tell}(x = n) \\
\mathit{Mutation} & = & \mathit{when}\ \mathit{trigger}\ \mathit{do} \\
 & & \quad \star\,!\,(\mathit{tell}(\mathit{mut} = 1) \parallel \mathit{next}\ \mathit{tell}(x = f_m)) \\
\mathit{Gene} & = & !\ \ \mathit{unless}\ \mathit{mut} = 1\ \mathit{next}\ \mathit{tell}(x = f_w) \\
\mathit{ControlRegion} & = & \mathit{Start} \parallel \mathit{Mutation} \parallel \mathit{Gene}
\end{array}
$$

A (very) simple bio model in NTCC

$$
\begin{array}{lcl}
\mathit{Start} & = & \mathit{tell}(x = n) \\
\mathit{Mutation} & = & \mathit{when}\ \mathit{trigger}\ \mathit{do} \\
 & & \quad \star\,!\,(\mathit{tell}(\mathit{mut} = 1) \parallel \mathit{next}\ \mathit{tell}(x = f_m)) \\
\mathit{Gene} & = & !\ \ \mathit{unless}\ \mathit{mut} = 1\ \mathit{next}\ \mathit{tell}(x = f_w) \\
\mathit{ControlRegion} & = & \mathit{Start} \parallel \mathit{Mutation} \parallel \mathit{Gene}
\end{array}
$$

when some triggering condition occurs

At some unspecified time in the future

- always the mutation signal is present $\mathit{tell}(\mathit{mut} = 1)$
- The number of interacting molecules is fixed to some f_m
- The gene can no longer interact with the usual quantities f_w

NTCC Semantics

Syntax

- Internal Transitions
  - Between configurations (process-store pairs)
    $\langle P, d\rangle \longrightarrow \langle P', d'\rangle$

Example

On $Store = Na_I > Na_{IDEAL}$

$$when\ Na_I > Na_{IDEAL}\ do\ tell(PhaseNa = 1)$$

$$\longrightarrow$$

$$\langle skip, PhaseNa = 1 \wedge Na_I > Na_{IDEAL}\rangle$$

NTCC Semantics

Syntax

- Internal Transitions
  - Between configurations (process-store pairs)
    $\langle P, d\rangle \longrightarrow \langle P', d'\rangle$
- Observable Transitions
  $P \xrightarrow{(c,d)} R$

Example

$$\mathit{when}\ Na_I > Na_{IDEAL}\ \mathit{do}\ (\ \mathit{next}\ \mathit{tell}(Na_I = Na'_I - 3) \parallel \mathit{tell}(PhaseNa = 1)\ )$$

$$(Na_I > Na_{IDEAL},\ PhaseNa = 1 \wedge Na_I > Na_{IDEAL}) \Longrightarrow \mathit{tell}(Na_I = Na'_I - 3) \quad (\mathit{Residual\ process})$$

A NTCC process run

- infinite sequence of observable transitions
  $P = P_1 \xrightarrow{(s_1,r_1)} P_2 \xrightarrow{(s_2,r_2)} P_3 \xrightarrow{(s_3,r_3)} \ldots$
- written as
  $P \xrightarrow{(\alpha,\alpha')}{}^{\omega}$
  where $\alpha = s_1.s_2.s_3 \ldots$ and $\alpha' = r_1.r_2.r_3 \ldots$

A NTCC process run

- infinite sequence of observable transitions
  $P = P_1 \xrightarrow{(s_1,r_1)} P_2 \xrightarrow{(s_2,r_2)} P_3 \xrightarrow{(s_3,r_3)} \ldots$
- written as
  $P \xrightarrow{(\alpha,\alpha')} \omega$
  where $\alpha = s_1.s_2.s_3 \ldots$ and $\alpha' = r_1.r_2.r_3 \ldots$

Srtrongest Postcondition

All output sequences that a process P can possibly output

$$sp(P) = \{\alpha' \mid \text{for some } \alpha : P \xrightarrow{(\alpha,\alpha')} \omega\}$$

Specification and Verification for NTCC Processes

- Constraints Linear Temporal Logic (CLTL)

$$F, G, \ldots := c \mid \dot{\mathtt{true}} \mid \dot{\mathtt{false}} \mid F \dot{\wedge} G \mid F \dot{\vee} G \mid \dot{\neg} F \mid \dot{\exists}_x F \mid \circ F \mid \Box F \mid \Diamond F$$

- Interpretations: infinite sequences of constraints

Specification and Verification for NTCC Processes

- Constraints Linear Temporal Logic (CLTL)

$$F, G, \ldots := c \mid \dot{\mathtt{true}} \mid \dot{\mathtt{false}} \mid F \dot{\wedge} G \mid F \dot{\vee} G \mid \dot{\neg} F \mid \dot{\exists}_x F \mid \circ F \mid \Box F \mid \Diamond F$$

- Interpretations: infinite sequences of constraints

Process P satisfies formula F

$P \models_{\mathrm{CLTL}} F$, iff $sp(P) \subseteq [\![F]\!]$

There is a proof system for temporal properties of NTCC programs

A proof system for NTCC

$$\text{LTELL}\quad \mathit{tell}(c) \vdash c \qquad \text{LCONS}\quad \frac{P \vdash A}{P \vdash B}\ \text{if } A \dot{\Rightarrow} B$$

$$\text{LPAR}\quad \frac{P \vdash A \quad Q \vdash B}{P \parallel Q \vdash A \dot{\wedge} B} \qquad \text{LUNL}\ \frac{P \vdash A}{\mathit{unless}\ c\ \mathit{next}\ P \vdash c \dot{\vee} \circ A}$$

$$\text{LREP}\ \frac{P \vdash A}{!P \vdash \Box A} \qquad \text{LLOC}\ \frac{P \vdash A}{\mathit{local}\ x\ \mathit{in}\ P \vdash \dot{\exists}_x A}$$

$$\text{LSTAR}\ \frac{P \vdash A}{\star P \vdash \Diamond A} \qquad \text{LNEXT}\ \frac{P \vdash A}{\mathit{next}\ P \vdash \circ A}$$

$$\text{LSUM}\ \frac{\forall i \in I \quad P_i \vdash A_i}{\sum_{i \in I} \mathit{when}\ c_i\ \mathit{do}\ P_i \vdash \dot{\bigvee}_{i \in I}(c_i \dot{\wedge} A_i) \dot{\vee} \dot{\bigwedge}_{i \in I} \dot{\neg} c_i}$$

Two ways to describe the behavior of a system

For some NTCC process,

- Run a simulation of its internal/external transitions
- Define its desirable properties and use the proof system to verify them

Using the Proof System

Property and Processes

"Under the triggering condition, the value of x is always determined by f_m"

$$(ControlRegion \parallel tell(trigger)) \vdash \Diamond\Box x = f_m$$

proof

Using the Proof System

Property and Processes

"Under the triggering condition, the value of x is always determined by f_m"

$$\begin{array}{lcl} \mathit{Start} & = & \mathit{tell}(x = n) \\ \mathit{Gene} & = & ! \ \mathit{unless}\ \mathit{mut} = 1\ \mathit{next}\ \mathit{tell}(x = f_w) \end{array}$$

proof

The formulae of processes *Start* and *Gene* :

$$\begin{array}{lcl} \mathit{Start} & \vdash & x = n \\ \mathit{Gene} & \vdash & \Box(\mathit{mut} = 1 \dot{\vee} \circ x = f_w) \end{array}$$

Using the Proof System

Property and Processes

"Under the triggering condition, the value of x is always determined by f_m"

$$Mutation \quad = \quad when\ trigger\ do \\ \star\,!\,(tell(mut = 1) \parallel next\ tell(x = f_m))$$

proof

So that :

$$Start \parallel Gene \vdash \ (\ x = n\ )\ \dot{\wedge}\ (\ \Diamond\Box(mut = 1 \dot{\wedge} \circ x = f_m)\ )$$

Using the Proof System

Property and Processes

"Under the triggering condition, the value of x is always determined by f_m"

$$Mutation \quad = \quad when\ trigger\ do \\ \star\,!\,(tell(mut = 1) \parallel next\ tell(x = f_m))$$

proof

The formulae of process $Mutation \parallel tell(trigger)$:

$$\begin{array}{lll} Mutation & \vdash & (trigger \mathbin{\dot{\wedge}} \Box(mut = 1 \mathbin{\dot{\vee}} \circ x = f_w)) \mathbin{\dot{\vee}} \dot{\neg}\, trigger \\ tell(trigger) & \vdash & trigger \end{array}$$

Using the Proof System

Property and Processes

"Under the triggering condition, the value of x is always determined by f_m"

$$(\mathit{ControlRegion} \parallel \mathit{tell}(\mathit{trigger})) \vdash \Diamond\Box x = f_m$$

proof

So that :

$$\mathit{Mutation} \parallel \mathit{tell}(\mathit{trigger}) \quad \vdash \quad \Box(\mathit{mut} = 1 \dot{\vee} \circ x = f_w)$$

Using the Proof System

Property and Processes

"Under the triggering condition, the value of x is always determined by f_m"

$$(ControlRegion \parallel tell(trigger)) \vdash \Diamond\Box x = f_m$$

proof

$$Gene \parallel Start \parallel Mutation \parallel tell(trigger) \vdash$$
$$\Box(mut = 1 \dot{\vee} \circ x = f_w)\ ) \ \dot{\wedge}\ (\ x = n\ ) \ \dot{\wedge}\ \Diamond\Box(mut = 1 \dot{\wedge} \circ x = f_m)$$

Deduction

Let $CR = ControlRegion \parallel tell(trigger)$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{CR \vdash \Box(mut = 1 \dot{\vee} \circ x = f_w) \dot{\wedge} (x = n) \dot{\wedge} (\Diamond\Box(mut = 1 \dot{\wedge} \ldots} \text{LC}}{CR \vdash \Box(mut = 1 \dot{\vee} \circ x = f_w) \dot{\wedge} \Diamond\Box(mut = 1 \dot{\wedge} \circ x = f_m)} \text{LC}}{CR \vdash \Diamond\Box((mut = 1 \dot{\vee} \circ x = f_w) \dot{\wedge} (mut = 1 \dot{\wedge} \circ x = f_m))} \text{LC}}{CR \vdash \Diamond\Box(mut = 1 \dot{\wedge} (mut = 1 \dot{\wedge} \circ x = f_m))} \text{LC}}{CR \vdash \Diamond\Box \circ x = f_m} \text{LC}}{CR \vdash \Diamond\Box x = f_m} \text{LC}$$

NTCC model of the Sodium pump

- Models active and passive ion transport
- Uses:
  - constraints for quantitative aspects (e.g. number of ions interchanged)
  - non determinism to model reversible reactions
  - temporal constructs to represent relative delays of interactions
  - constraints as signals to be used in proving properties of the system

Active transport

$$
\begin{aligned}
&\mathit{NaPhase1} \;=\\
&\quad \mathit{when}\ (\mathit{Na}_I > \mathit{Na}_{th} \vee \mathit{K}_I < \mathit{K}_{th}) \wedge \mathit{Pump} = \texttt{Empty} \wedge \mathit{OPump} = \texttt{In}\ )\\
&\qquad\quad \mathit{next}\ (\ \mathit{Na}_I := \mathit{Na}_I - 3 \parallel \mathit{Pump} := \texttt{Na}\ \parallel \mathit{NaPhase2}\ )\\
&\quad + \ \mathit{next}\ \mathit{NaPhase1} \parallel \mathit{tell}(\mathit{unchanged}(\mathit{K})) \parallel \mathit{tell}(\mathit{unchanged}(\mathit{Na}))
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{NaPhase2} \;=\\
&\quad \mathit{when}\ (\mathit{Pump} = \texttt{Na} \wedge \mathit{Alpha} = \texttt{free} \wedge \mathit{ATP} > 0)\\
&\qquad \mathit{next}\ (\mathit{OPump} := \texttt{Out} \parallel \mathit{Alpha} := \texttt{P} \parallel \mathit{ADP} := 1\\
&\qquad\quad \parallel \mathit{tell}(\mathit{unchanged}(\mathit{K})) \parallel \mathit{tell}(\mathit{unchanged}(\mathit{Na})) \parallel \mathit{NaPhase3}\ )\\
&+ \mathit{next}\ \mathit{NaPhase2} \parallel \mathit{tell}(\mathit{unchanged}(\mathit{K})) \parallel \mathit{tell}(\mathit{unchanged}(\mathit{Na})))
\end{aligned}
$$

Passive transport

$$
\begin{aligned}
&\mathit{PassiveNa}_d \;=\\
&\;\mathit{unless}\ \mathit{Na}_O = \mathit{Na}_I\ \mathit{next}\\
&\quad(\mathit{next}^{\,d}\,\mathit{PassiveNa}\\
&\quad\|\ \star_{[0,d]}(\mathit{unless}\ \mathit{unchanged}(\mathit{Na})\ \mathit{next}\\
&\qquad\qquad(\mathit{Na}_I := \mathit{Na}_I + 3 \parallel \mathit{Na}_O := \mathit{Na}_O - 3)\\
&\qquad\|\ \mathit{when}\ \mathit{unchanged}(\mathit{Na})\ \mathit{do}\\
&\qquad\qquad(\mathit{Na}_I := \mathit{Na}_I + 3 \parallel \mathit{Na}_O := \mathit{Na}_O - 3))
\end{aligned}
$$

The System

$$
\begin{aligned}
&\mathit{Control} \;=\\
&\;!\;(\mathit{when}\; Na_I = Na_O\; \mathit{do}\; \mathit{tell}(\mathit{equil}(Na)) \parallel\\
&\quad \mathit{when}\; K_I = K_O\; \mathit{do}\; \mathit{tell}(\mathit{equil}(K)) \parallel\\
&\quad \mathit{when}\; \mathit{equil}(Na) \vee \mathit{equil}(K) \vee M\; \mathit{do}\; !\; \mathit{tell}(\mathit{death}))
\end{aligned}
$$

$$
\begin{aligned}
\mathit{System} \;=\; &\mathit{local}\; x_1, ..., x_k\; \mathit{in}\\
&\mathit{Start}(x_1, ..., x_k) \parallel \mathit{ActiveTrans} \parallel \mathit{PassiveTrans}\\
&\parallel \mathit{Control} \parallel \mathit{Environment}
\end{aligned}
$$

Modeling disfunction

$$\textit{Environment} = \star_{[m,n]} \textit{when}\ \textit{Alpha} = \texttt{free}\ \textit{do}\ !\ \textit{Alpha} := \texttt{null}$$

So the following can be proved:
$\textit{System} \vdash \diamondsuit\Box\ \textit{death}$

Future Work

- Tackling more comprehensive biological models
- Fully integrating stochastic behavior
- releasing an efficient NTCC interpreter for biological processes simulation.

Thanks