

*Probabilistic Models for
Concurrency
and
A Potential Application*

MICHAEL W. MISLOVE*

Tulane University

LIAFA

July, 2001

* WORK PARTIALLY SUPPORTED BY THE NSF AND THE
ONR.

Outline of talk:

Goal: Describe progress in devising a model for concurrency that supports both nondeterminism and probabilistic choice, and also outline one of its potential applications.

- Review untimed *CSP*
- Give a specification of a simple buffer in *CSP*
- Consider extension to *PCSP* - probabilistic *CSP* – to include lossy channel
- Show why this model is not satisfactory
- Describe better model
- Recall basics of Hybrid Systems
- Outline method for translating hybrid systems into *PCSP*
- Describe quantified temporal logic for analyzing processes in *PCSP*

CSP : process algebra for specifying and verifying concurrent processes.

$$P ::= STOP \mid SKIP \mid a \rightarrow P \mid P \setminus B \mid P; Q \\ \mid P \underset{C}{\parallel} Q \mid P \sqcap Q \mid P \sqcup Q \mid X \mid \text{rec } X.P$$

<i>STOP</i>	Deadlock
<i>SKIP</i>	normal termination
$a \rightarrow P$	first execute $a \in A$, then act like P
$P \setminus B$	P with all actions in $B \subseteq A$ hidden
$P; Q$	sequential composition
$P \underset{C}{\parallel} Q$	P and Q synchronizing on actions in C
$P \sqcap Q$	internal choice
$P \sqcup Q$	external choice
X	process variable
$\text{rec } X.P$	recursion

- Both a process and its specification can be written in *CSP*.
- Processes understood in terms of the *events* they participate in.

Failures-divergences model for CSP:

$$\mathbb{FD} = \{(F, D) \mid F = \{(s, X) \mid s \text{ trace of } P \ \& \ P \text{ refuses } X \text{ after } s\} \cup \{(s \hat{t}, X) \mid P \text{ diverges on } s\} \\ D = \{s \hat{t} \mid P \text{ diverges on } s\} \}$$

$$\llbracket a \rightarrow STOP \rrbracket \equiv$$

$$(\{(\langle \rangle, X) \mid a \notin X\} \cup \{(a, X) \mid X \subseteq A\}, \emptyset).$$

$$\llbracket a \rightarrow (\text{rec } X.a \rightarrow X) \setminus a \rrbracket =$$

$$(\{(\langle \rangle, X) \mid a \notin X\} \cup \{(as, X) \mid s \in A^*, X \subseteq A\}, aA^*)$$

- Process P *refines* process Q iff every behavior of P is also a behavior of Q .
- Process P *satisfies* specification S iff every behavior of P is allowed by S .

$$S = (F_1, D_1) \sqsubseteq (F_2, D_2) = P \Leftrightarrow F_2 \subseteq F_1 \ \& \ D_2 \subseteq D_1$$

- Reduces verification to set containment.
- Automated support is available to check this.

A Simple Buffer

A *buffer* for data of type T should:

- (i) Only input on channel *in* and output on channel *out*. It correctly copies its input to its output without loss of data or reordering.
- (ii) Always accept input when empty.
- (iii) Always be willing to output when non-empty.

Translation into *CSP*: B is a buffer if:

$$(i) \quad s \in \text{traces}(B) \Rightarrow s \in (\text{in}.T \cup \text{out}.T)^* \\ \wedge s \downarrow \text{out} \leq s \downarrow \text{in}.$$

$$(ii) \quad (s, X) \in \text{failures}(B) \wedge s \downarrow \text{in} = s \downarrow \text{out} \Rightarrow \\ X \cap \text{in}.T = \emptyset.$$

$$(iii) \quad (s, X) \in \text{failures}(B) \wedge s \downarrow \text{out} < s \downarrow \text{in} \Rightarrow \\ \text{out}.T \not\subseteq X.$$

For example,

$$COPY := \text{in}?x \rightarrow \text{out}!x \rightarrow COPY$$

satisfies this specification.

An N -place buffer can be specified as $BUFF_{\langle \rangle}^N$, where:

$$\begin{aligned}
 BUFF_{\langle \rangle}^N &:= in?x \rightarrow BUFF_{\langle x \rangle}^N \\
 BUFF_{t^{\sim}\langle a \rangle}^N &:= (in?x \rightarrow BUFF_{\langle x \rangle^{\sim}t^{\sim}\langle a \rangle}^N) \\
 &\quad \triangleleft \#t < N - 1 \triangleright \\
 &\quad ((out!a \rightarrow BUFF_t^N) \sqcap STOP) \\
 &\quad \square \\
 &\quad (out!a \rightarrow BUFF_t^N).
 \end{aligned}$$

Probabilistic CSP:

A la Morgan, McIver, Sanders and Seidel.

- Adds probabilistic choice operators $P \dot{+} Q$ to un-timed CSP.
- Built on top of failures-divergences model for CSP using standard domain construction.

$$\mathcal{P}_{Pr}(\mathbb{FD}) = \{\mu \mid \mu \text{ probability measure on } \mathbb{FD}\}$$

- Not all expected laws hold. For example

$$P \sqcap P \neq P \quad P \square P \neq P$$

- Unwanted laws do hold. For example:

$$(P \dot{+} Q) \sqcap R = (P \sqcap R) \dot{+} (Q \sqcap R).$$

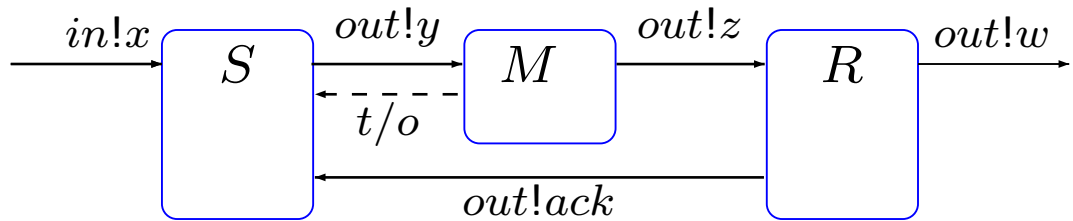
Leads to unexpected reasoning:

$$(P_{1/3} \dot{+} Q) \sqcap (P_{1/3} \dot{+} Q) = (P_{1/3} \dot{+} Q)_{1/3} \dot{+} (P \sqcap Q)$$

has probability $1/9 \leq p \leq 2/3$ of acting like P .

Communication over a lossy medium

PCSP model of Stop and Wait Protocol



$$S := in?x \rightarrow S'$$

$$S' := out!y \rightarrow S''$$

$$S'' := (timeout \rightarrow S') \square (in?ack \rightarrow S)$$

$$M := (timeout \rightarrow M) \overset{r}{+} (in?y \rightarrow out!z \rightarrow M)$$

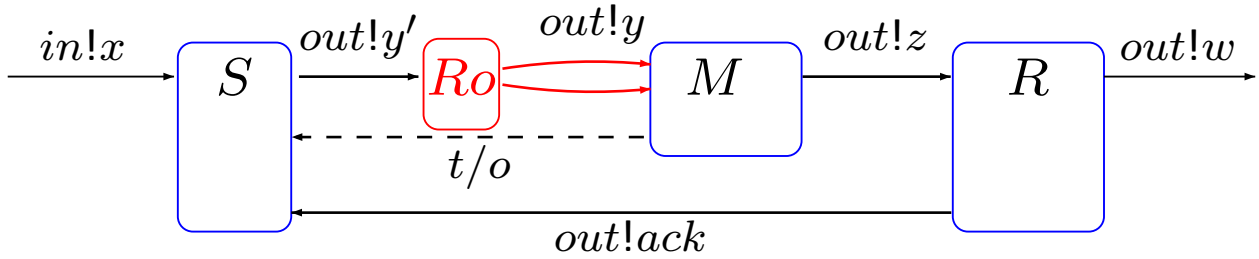
$$R := in?z \rightarrow out!w \rightarrow out!ack \rightarrow R$$

$$P := (S \underset{B}{\parallel} M \underset{B}{\parallel} R) \setminus C \quad B = C \cup \{w, x\}, C = \{y, z, ack\}$$

$$\simeq \text{rec } X.(in?x \rightarrow out!x \rightarrow X)$$

and has probability r of losing the message on any run.

But, what if we add a router between S and the M ?



$$S := in?x \rightarrow S'$$

$$S' := out!y' \rightarrow S''$$

$$S'' := (timeout \rightarrow S') \sqcap (in?ack \rightarrow S)$$

$$Ro := (in?y' \rightarrow (out?y_1 \sqcap out!y_2)) \rightarrow Ro$$

$$M := (t/o \rightarrow M) \text{ r+ } ((in?y_1 \sqcap in?y_2) \rightarrow out!z \rightarrow M)$$

$$R := in?z \rightarrow out!w \rightarrow out!ack \rightarrow R$$

$$P := (S \underset{D}{\parallel} (Ro \underset{D}{\parallel} M) \underset{D}{\parallel} R) \setminus E$$

$$\simeq (S \underset{D}{\parallel} (M \sqcap M) \underset{D}{\parallel} R) \setminus E$$

has probability $r^2 \leq p \leq (1 - r)$ of failure.

An Alternative Approach

Joint work with Gavin Lowe (Oxford)

$P ::= STOP \mid SKIP \mid a \rightarrow P \mid P \overset{r}{+} Q \mid P \overset{c}{\parallel} Q \mid P \sqcap Q \mid P \sqcup Q \mid X \mid \text{rec } X.P$

Denotational model:

$$F \simeq 1 \oplus \mathcal{P}_{CC}(\mathcal{P}_{Pr}(A \multimap F)) = \lim_n (1 \oplus \mathcal{P}_{CC} \circ \mathcal{P}_{Pr})^n(1),$$

$$F_1 = \{1\}, F_2 = 1 \oplus \mathcal{P}_{CC}(\mathcal{P}_{Pr}(A \multimap 1)) \dots$$

Processes are:

STOP or...

...members of a special power domain...

...over probabilistic choices of...

...(finite) partial functions from...

... A to processes

For $a_1, a_2 \in A$, processes P_1, P_2 , & $0 \leq r \leq 1$

$$\llbracket (a_1 \rightarrow P_1) \overset{r}{+} (a_2 \rightarrow P_2) \rrbracket = \langle \delta_{a_1 \rightarrow \llbracket P_1 \rrbracket} \overset{r}{+} \delta_{a_2 \rightarrow \llbracket P_2 \rrbracket} \rangle$$

Laws in our model:

Nondeterministic laws:

$$P \sqcap P = P, P \sqcap Q = Q \sqcap P, \\ (P \sqcap Q) \sqcap R = P \sqcap (Q \sqcap R).$$

$$P \sqcup P = P, P \sqcup Q = Q \sqcup P, \\ (P \sqcup Q) \sqcup R = P \sqcup (Q \sqcup R).$$

A \sqcap -semilattice and a \sqcup -semilattice.

Probabilistic laws: $\{r \vdash \mid 0 \leq r \leq 1\}$ satisfy:

- $P \ r \vdash P = P$
- $P \ r \vdash Q = Q \ 1-r \vdash P$
- $P \ 1 \vdash Q = P$
- $(P \ r \vdash Q) \ s \vdash R = P \ rs \vdash (Q \ \frac{s(1-r)}{1-rs} \vdash R)$

(if $rs < 1$)

A probabilistic algebra.

Laws that *don't* hold:

$$a \rightarrow (P \sqcap Q) \neq (a \rightarrow P) \sqcap (a \rightarrow Q)$$

$$a \rightarrow (P \text{ r+ } Q) \neq (a \rightarrow P) \text{ r+ } (a \rightarrow Q)$$

$$(P \text{ r+ } Q) \parallel_C R \neq (P \parallel_C R) \text{ r+ } (Q \parallel_C R)$$

For example:

$$(P \text{ r+ } Q) \parallel_C (d \rightarrow (P \sqcap Q))$$

should equal

$$d \rightarrow ((P \text{ r+ } Q) \parallel_C (P \sqcap Q))$$

if $d \notin C$. Instead,

$$(P \text{ r+ } Q) \parallel_C R = (P \parallel_C R) \text{ r+ } (Q \parallel_C R)$$

implies $(P \text{ r+ } Q) \parallel_C (d \rightarrow (P \sqcap Q))$ resolves $P \text{ r+ } Q$ *before* $P \sqcap Q$ even if $d \notin C$.

Moral:

$$a \rightarrow - \quad \text{and} \quad \parallel_C$$

must be defined at the \mathcal{P}_{CC} -level in

$$\mathcal{P}_{CC}(\mathcal{P}_{Pr}(A \rightarrow F)).$$

But, in our model,

$$\begin{aligned} P & := (S \parallel_D (Ro \parallel_D M) \parallel_D R) \setminus E \\ & \simeq (S \parallel_D (M \sqcap M) \parallel_D R) \setminus E \\ & = (S \parallel_D M \parallel_D R) \setminus E \end{aligned}$$

has probability r of failure.

Work to be done:

- Validate the denotational model.
- Devise operational model and prove congruence theorem.
- Devise logic for reasoning about processes.

Hybrid Systems

Variables - $(x_1, \dots, x_n) \in \mathbb{R}^n$, $n = \text{dimension}$.

Control graph: A finite, directed multigraph (V, E) . The nodes of V are the *locations*.

State space: $V \times \mathbb{R}^n$.

Initial, invariant and flow conditions:

- *init* defines a set of *initial states*.
 - *inv* assigns *invariant region* to each location.
 - *flow* assigns *flow conditions* to each location.
- These govern the behavior of the real variables for that location.

Jump conditions: Conditions on (x_1, \dots, x_n) under which jump is *enabled*.

Events: *event*: $E \rightarrow \Sigma$ labels jumps.

A Thermostat

- variable $x \in \mathbb{R}$ temperature
- locations *Off* and *On*

Initial state:

- *Off*: $x = 20$

Invariant regions:

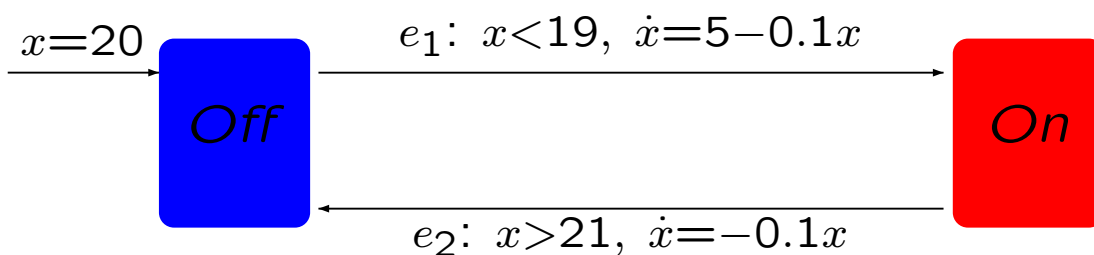
- *Off*: $x \geq 18$
- *On*: $x \leq 22$

Flow conditions:

- *Off*: $\dot{x} = -0.1x$
- *On*: $\dot{x} = 5 - 0.1x$

Enabled regions:

- *Off*: $x < 19$
- *On*: $x > 21$



Basic questions

Safety - What regions are *reachable*?

Liveness - Are there *infinite runs*?

Standard Approach: Devise discrete abstractions with equivalent behavior, and for which questions are decidable.

Language equivalences preserve *LTL* properties.

Bisimulation equivalences preserve *CTL* properties.

Hybrid Systems

A system is *initialized* if, whenever a jump changes the flow condition for a variable, then the jump reinitializes the variable.

An automaton is *rectangular* if all invariant and flow regions are products of intervals defined by rational numbers.

A rectangular automaton is *multirate* if

- each location has at most one initial state,
- jumps are deterministic, and
- flows are constant at each location.

A *timed automaton* is a multirate automaton for which $\dot{x} = 1$ for all variables x .

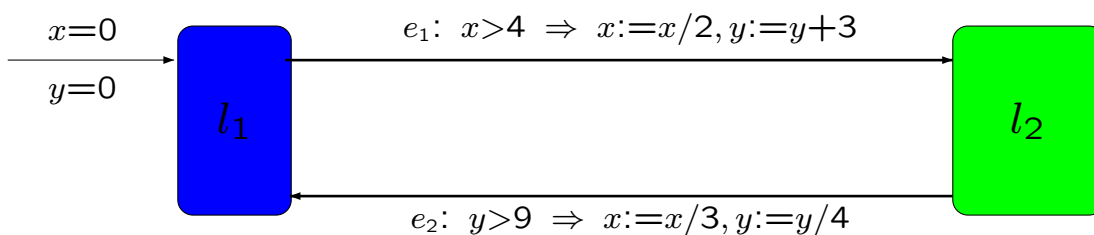
A simple example of a timed automaton T .

Invariant regions:

- $l_1: x, y \in [0, 5)$
- $l_2: x, y \in [0, 10)$

Flow conditions:

- $\dot{x} = \dot{y} = 1$.



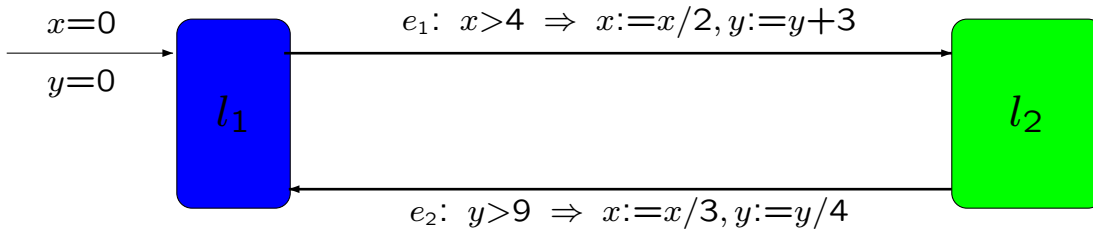
Hybrid Systems

Theorems:

- [Alur & Dill] The reachability problem and the ω -language emptiness problem for timed automata are decidable.
- [Henzinger, et al] LTL and CTL reachability problems for initialized multirate automata are decidable subject to the region in question being either a location or a rectangular set.
- [Henzinger, et al] The reachability problem for uninitialized multirate automata with one stopwatch and all other variables clocks is undecidable.

Question: How can one analyze hybrid systems for which these problems are undecidable?

From Hybrid Systems to PCSP



Defining a probabilistic approximation to T :

1) Partition $[0, 5)$ into: $x_1 = [0, 3)$, $x_2 = [3, 4]$, $x_3 = (4, 4.5)$, $x_4 = [4.5, 4.8)$ and $x_5 = [4.8, 5)$.

2) Partition $[0, 10)$ into: $y_1 = [0, 5]$, $y_2 = (5, 9]$, $y_3 = (9, 9.7)$ and $y_4 = [9.7, 10)$.

3) Use the flow conditions to assign probabilities to each subregion: these are clocks, so we use normalized Lebesgue measure on each interval:

<i>Location and Variables</i>	<i>Interval</i>	<i>Probability</i>	<i>Jumps</i>
l_1 & x -subintervals:	$[0, 3)$.6	a_1
	$[3, 4]$.2	a_2
	$(4, 4.5)$.1	a_3
	$[4.5, 4.8)$.06	a_4
	$[4.8, 5)$.04	a_5
l_2 & y -subintervals:	$[0, 5)$.5	b_1
	$[5, 9]$.4	b_2
	$(9, 9.7)$.07	b_3
	$[9.7, 10)$.03	b_4

4) Define two subprocesses:

P – representing l_1 , and Q – representing l_2 .

Further refine P and Q into subprocesses:

There are five representing restrictions of P :

$P_1 = P|_{[0,5)}$, $P_2 = P|_{[3,5)}$, $P_3 = P|_{(4,5)}$,
 $P_4 = P|_{[4.5,5)}$ and $P_5 = P|_{[4.8,5)}$.

Similarly, there are four subprocesses for Q :

$Q_1 = Q|_{[0,10)}$, $Q_2 = Q|_{[5,10)}$, $Q_3 = Q|_{[9,10)}$ and
 $Q_4 = Q|_{[9.7,10)}$.

5) Define mutually recursive equations:

$$P := (a_1 \rightarrow P_2) .6\uparrow ((a_2 \rightarrow P_3) .5\uparrow ((a_3 \rightarrow (P_4 \sqcap Q_2))) .5\uparrow ((a_4 \rightarrow (P_5 \sqcap Q_6)) .6\uparrow (a_5 \rightarrow Q_6))))$$

$$Q := (b_1 \rightarrow Q_2) .5\uparrow ((b_2 \rightarrow Q_3) .8\uparrow ((b_3 \rightarrow (Q_4 \sqcap P_3)) .35\uparrow (b_4 \rightarrow P_4)))$$

6) The *PCSP* process R that approximates T is the solution to the system in 5).

From Hybrid Systems to PCSP

Theorem [Alvarez-Manilla]

If X is locally compact, then *any* probability distribution μ on X is the supremum of an increasing family of simple measures $\sum_{i=1}^n r_i \delta_{x_i}$, where $\sum_i r_i = 1$.

Corollary Any probability distribution on X can be approximated arbitrarily closely by *PCSP* processes.

Partition X so that each x_i is in a unique element, then use procedure of previous slide.

*In other words, given probability distributions on each invariant region representing the flow conditions, we can realize the behavior as a limit of *PCSP*-processes.*

Question: Is this limit a process?

How to analyze PCSP?

Formulation of *CTL* in terms of predicate transformers:

$$\begin{array}{ll} S & - \text{ states,} \\ \mathcal{P}(S) = \{0, 1\}^S & - \text{ predicates over } S, \\ \text{prog}: \mathcal{P}(S) \rightarrow \mathcal{P}(S) & - \text{ predicate transformers - programs.} \\ & \text{prog}(A) = \text{wp.prog.A} \end{array}$$

Healthiness conditions:

$$\begin{array}{lll} \text{prog}(\emptyset) = \emptyset & & \text{excluded miracle} \\ A \subseteq B \Rightarrow \text{prog}(A) \subseteq \text{prog}(B) & & \text{monotone} \\ \text{prog}(A \cap B) = \text{prog}(A) \cap \text{prog}(B) & & \text{positively conjunctive} \end{array}$$

Modal operators for *CTL*

$$\begin{array}{lll} \text{Next} & - & \circ A := \text{prog}(A) \\ \text{Eventually} & - & \diamond A := \mu X.(A \cup \circ X) \quad \text{least fixed point} \\ \text{Always} & - & \square A := \nu X.(A \cap \circ X) \quad \text{greatest fixed point} \\ \text{Unless} & - & A \triangleright B := \nu X.B \cup (A \cap \circ X) \end{array}$$

Theorem:

System satisfies axioms that are complete for standard branching time temporal logic.

For example,

$$\square(A \Rightarrow B) \Rightarrow (\square A \Rightarrow \square B) \text{ and } (\square A \Rightarrow B) \wedge \diamond A \Rightarrow \diamond B$$

How to analyze PCSP?

McIver and Morgan: *Quantitative* temporal logic *qTL* for reasoning about *probabilistic processes with demonic nondeterminism* in terms of *expectation transformers*:

S - states,
 $\mathcal{E}(S) = [0, 1]^S$ - expectations over S ,
 $prog: \mathcal{E}(S) \rightarrow \mathcal{E}(S)$ - expectation transformers
- *programs*.

For deterministic $prog: S \rightarrow [0, 1]^S$, $s_0 \in S$ and $X \subseteq S$, $prog(s_0)(X) \in [0, 1]$

For deterministic $prog: S \rightarrow [0, 1]^S \wedge E \in \mathcal{E}(S)$,

$$prog(E) = \int E \, d.prog(-): S \rightarrow [0, 1] \quad \text{Kozen, 1983}$$

For nondeterministic $prog$

$$prog(E) = \sqcap_{prog \leq \mu} \int E \, d.\mu$$

How to analyze PCSP?

How to compare expectations:

$E \Rightarrow F$ - everywhere no more than

$E \Leftarrow F$ - everywhere no less than

$E \equiv F$ - everywhere equal.

Healthiness conditions:

$prog(E) \Rightarrow \sqcup E$ excluded miracle

$E \Rightarrow F \Rightarrow prog(E) \Rightarrow prog(F)$ monotone

$prog(E \text{ r+ } F) = prog(E) \text{ r+ } prog(F)$ subdistributivity

Modal operators for qTL

Next - $\circ E := prog(E)$

Eventually - $\diamond E := \mu X.(E \vee \circ X)$ least fixed point

Always - $\square E := \nu X.(E \wedge \circ X)$ greatest fixed point

Unless - $E \triangleright F := \nu X.F \vee (E \wedge \circ X)$

How to analyze PCSP?

Theorem: [Vardi's 0-1 Law]

Almost certain properties in linear temporal logic depend only on the probabilities being strictly between 0 and 1.

Morgan and McIver obtain 0–1 Laws for their logic. For example:

1) If $0 < r < 1$

$$\text{rec } X.(a \rightarrow X) \text{ } r\text{ } \dagger \text{ } (b \rightarrow X)$$

has probability 1 of executing a and probability 1 of executing b

2) $p_n = \text{rec } X.(a \rightarrow \text{STOP}) \text{ } n\text{ } \dagger \text{ } ((n := n+1) \rightarrow X)$

has probability $\frac{1}{n}$ of executing a .

Summary

We have described

- A model for probabilistic *CSP* that validates desirable laws,
- A method for translating hybrid systems into *PCSP* processes,
- A quantified temporal logic for reasoning about *PCSP* processes that also has 0–1 Laws.

Goal: Analyze hybrid systems using these tools to gain insights into their behavior.

In particular, can we make *almost sure assertions* about reachability and liveness using this approach?