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
\textit{CSP} : process algebra for specifying and verifying concurrent processes.

\[
P ::= \text{STOP} \mid \text{SKIP} \mid a \rightarrow P \mid P \setminus B \mid P;Q \mid P || Q \mid P \cap Q \mid P \bowtie Q \mid X \mid \text{rec } X.P
\]

- \text{STOP}\quad \text{Deadlock}
- \text{SKIP}\quad \text{normal termination}
- a \rightarrow P\quad \text{first execute } a \in A, \text{ then act like } P
- P \setminus B\quad P \text{ with all actions in } B \subseteq A \text{ hidden}
- P;Q\quad \text{sequential composition}
- P || Q\quad P \text{ and } Q \text{ synchronizing on actions in } C
- P \cap Q\quad \text{internal choice}
- P \bowtie Q\quad \text{external choice}
- X\quad \text{process variable}
- \text{rec } X.P\quad \text{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:**

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

\[[a \rightarrow STOP] \equiv \]

\[
\{(\langle \rangle, X) \mid a \not\in X\} \cup \{(a, X) \mid X \subseteq A\}, \emptyset \).
\]

\[[a \rightarrow \text{rec } X.a \rightarrow X) \setminus a] = \]

\[
\{(\langle \rangle, X) \mid a \not\in X\} \cup \{(as, X) \mid s \in A^*, X \subseteq A\}, aA^*)\]
– 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 $S$.

\[ S = (F_1, D_1) \subseteq (F_2, D_2) = P \iff F_2 \subseteq F_1 \ \& \ D_2 \subseteq D_1 \]

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

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

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

For example,

$$COPY := in?x \rightarrow out!x \rightarrow COPY$$

satisfies this specification.
An $N$-place buffer can be specified as $BUFF^N_{\langle \rangle}$, where:

$$BUFF^N_{\langle \rangle} ::= \text{in}\?x \rightarrow BUFF^N_{\langle x \rangle}$$

$$BUFF^N_{t\langle a \rangle} ::= (\text{in}\?x \rightarrow BUFF_{t\langle x \rangle}^{N - t\langle a \rangle})$$

$$\langle \#t < N - 1 \rangle$$

$$((\text{out}\!a \rightarrow BUFF_t^N) \sqcap \text{STOP})$$

$$\square$$

$$((\text{out}\!a \rightarrow BUFF_t^N)).$$
Probabilistic CSP:

A la Morgan, McIver, Sanders and Seidel.

- Adds probabilistic choice operators $P + Q$ to untimed CSP.
- Built on top of failures-divergences model for CSP using standard domain construction.
  \[ \mathcal{P}_{Pr}(FD) = \{ \mu \mid \mu \text{ probability measure on } FD \} \]
- Not all expected laws hold. For example
  \[ P \cap P \neq P \quad P \oplus P \neq P \]
- Unwanted laws do hold. For example:
  \[ (P + Q) \cap R = (P \cap R) + (Q \cap R). \]
  Leads to unexpected reasoning:
  \[ (P_{1/3} + Q) \cap (P_{1/3} + Q) = (P_{1/3} + Q)_{1/3} + (P \cap Q) \]
  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

\[ S := \text{in}!x \rightarrow S' \]
\[ S' := \text{out}!y \rightarrow S'' \]
\[ S'' := (\text{timeout} \rightarrow S') \square (\text{in}!\text{ack} \rightarrow S) \]
\[ M := (\text{timeout} \rightarrow M) \text{r+} (\text{in}!y \rightarrow \text{out}!z \rightarrow M) \]
\[ R := \text{in}!z \rightarrow \text{out}!w \rightarrow \text{out}!\text{ack} \rightarrow R \]

\[ P := (S \parallel M \parallel R) \backslash C \quad B = C \cup \{w, x\}, C = \{y, z, \text{ack}\} \]
\[ \simeq \text{rec } X. (\text{in}!x \rightarrow \text{out}!x \rightarrow X) \]

and has probability \( r \) of losing the message on any run.
But, what if we add a router between $S$ and the $M$?

$$S := \text{in}?!x \rightarrow S'$$
$$S' := \text{out}?!y' \rightarrow S''$$
$$S'' := (\text{timeout} \rightarrow S') \boxdot (\text{in}?!\text{ack} \rightarrow S)$$
$$Ro := (\text{in}?!y' \rightarrow (\text{out}?!y_1 \sqcap \text{out}?!y_2)) \rightarrow Ro$$
$$M := (t/o \rightarrow M) \mathbin{\mathbin{+}} ((\text{in}?!y_1 \Box \text{in}?!y_2) \rightarrow \text{out}?!z \rightarrow M)$$
$$R := \text{in}?!z \rightarrow \text{out}?!w \rightarrow \text{out}?!\text{ack} \rightarrow R$$

$$P := (S \parallel (Ro \parallel M) \parallel R) \setminus E$$
$$\simeq (S \parallel (M \sqcap M) \parallel R) \setminus E$$

has probability $r^2 \leq p \leq (1 - r)$ of failure.
An Alternative Approach
Joint work with Gavin Lowe (Oxford)

\[ P ::= \text{STOP} \mid \text{SKIP} \mid a \rightarrow P \mid P \mathbin{||} Q \mid P \mathbin{\parallel} Q \mid P \mathbin{\square} Q \mid X \mid \text{rec } X.P \]

Denotational model:

\[ F \simeq 1 \oplus \mathcal{P}_{CC}(\mathcal{P}_{Pr}(A \rightarrow F)) \equiv \lim_n (1 \oplus \mathcal{P}_{CC} \circ \mathcal{P}_{Pr})^n(1), \]

\[ F_1 = \{1\}, F_2 = 1 \oplus \mathcal{P}_{CC}(\mathcal{P}_{Pr}(A \rightarrow 1)) \ldots \]

Processes are:
\[ \text{STOP} \] 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 \( P_1, P_2 \), \& \( 0 \leq r \leq 1 \)

\[ \semantics{(a_1 \rightarrow P_1) \mathbin{\top} (a_2 \rightarrow P_2)} = \langle \delta_{a_1 \rightarrow \semantics{P_1}} \mathbin{\top} \delta_{a_2 \rightarrow \semantics{P_2}} \rangle \]
Laws in our model:

**Nondeterministic laws:**

\[ P \sqcap P = P, \quad P \sqcap Q = Q \sqcap P, \]
\[ (P \sqcap Q) \sqcap R = P \sqcap (Q \sqcap R). \]

\[ P \Box P = P, \quad P \Box Q = Q \Box P, \]
\[ (P \Box Q) \Box R = P \Box (Q \Box R). \]

A \( \sqcap \)-semilattice and a \( \Box \)-semilattice.

**Probabilistic laws:** \( \{ r^+ \mid 0 \leq r \leq 1 \} \) satisfy:

1. \( P \; r^+ + P = P \)
2. \( P \; r^+ + Q = Q \; 1-r^+ + P \)
3. \( P \; 1^+ + Q = P \)
4. \( (P \; r^+ + Q) \; s^+ + R = P \; rs^+ + (Q \; s(1-r) + R) \frac{1}{1-rs} \)

(if \( rs < 1 \))

A probabilistic algebra.
Laws that don’t hold:

\[
\begin{align*}
a \rightarrow (P \cap Q) & \neq (a \rightarrow P) \cap (a \rightarrow Q) \\
a \rightarrow (P \vdash Q) & \neq (a \rightarrow P) \vdash (a \rightarrow Q) \\
(P \vdash Q) \parallel R & \neq (P \parallel R) \vdash (Q \parallel R)
\end{align*}
\]

For example:

\[
(P \vdash Q) \parallel_C (d \rightarrow (P \cap Q))
\]

should equal

\[
d \rightarrow ((P \vdash Q) \parallel_C (P \cap Q))
\]

if \( d \notin C \). Instead,

\[
(P \vdash Q) \parallel_C R = (P \parallel_C R) \vdash (Q \parallel_C R)
\]

implies \((P \vdash Q) \parallel_C (d \rightarrow (P \cap Q))\) resolves \( P \vdash Q \) before \( P \cap Q \) 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,
\[
\begin{align*}
P & := (S \parallel (R_{o} \parallel M) \parallel R) \backslash E \\
& \simeq (S \parallel (M \cap M) \parallel R) \backslash E \\
& = (S \parallel M \parallel R) \backslash E
\end{align*}
\]
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 - \((x_1, \ldots, x_n) \in \mathbb{R}^n, n = \text{dimension}\).

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

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

Events: \textit{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:
- $\dot{x} = \dot{y} = 1$.

```
Invariant regions:
- l1: x, y \in [0, 5)
- l2: x, y \in [0, 10)
Flow conditions:
- \dot{x} = \dot{y} = 1.
```

\[ x=0 \quad y=0 \]
\[ l_1 \rightarrow l_2 \]
\[ e_1: x>4 \Rightarrow x:=x/2, y:=y+3 \]
\[ e_2: y>9 \Rightarrow x:=x/3, y:=y/4 \]
Hybrid Systems

Theorems:
- [Alur & Dill] The reachability problem and the \( \omega \)-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

\[ x=0 \quad e_1: \ x>4 \Rightarrow x:=x/2, y:=y+3 \]
\[ y=0 \quad e_2: \ y>9 \Rightarrow x:=x/3, y:=y/4 \]

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:

<table>
<thead>
<tr>
<th>Location and Variables</th>
<th>Interval</th>
<th>Probability</th>
<th>Jumps</th>
</tr>
</thead>
<tbody>
<tr>
<td>( l_1 ) &amp; ( x )-subintervals:</td>
<td>([0, 3))</td>
<td>.6</td>
<td>( a_1 )</td>
</tr>
<tr>
<td></td>
<td>([3, 4])</td>
<td>.2</td>
<td>( a_2 )</td>
</tr>
<tr>
<td></td>
<td>((4, 4.5))</td>
<td>.1</td>
<td>( a_3 )</td>
</tr>
<tr>
<td></td>
<td>([4.5, 4.8))</td>
<td>.06</td>
<td>( a_4 )</td>
</tr>
<tr>
<td></td>
<td>([4.8, 5))</td>
<td>.04</td>
<td>( a_5 )</td>
</tr>
<tr>
<td>( l_2 ) &amp; ( y )-subintervals:</td>
<td>([0, 5))</td>
<td>.5</td>
<td>( b_1 )</td>
</tr>
<tr>
<td></td>
<td>([5, 9])</td>
<td>.4</td>
<td>( b_2 )</td>
</tr>
<tr>
<td></td>
<td>((9, 9.7))</td>
<td>.07</td>
<td>( b_3 )</td>
</tr>
<tr>
<td></td>
<td>([9.7, 10))</td>
<td>.03</td>
<td>( b_4 )</td>
</tr>
</tbody>
</table>
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 \( P \):
\[
P_1 = P|_{[0,5)}, \ P_2 = P|_{[3,5)}, \ P_3 = P|_{(4,5)}, \\
P_4 = P|_{[4.5,5)} \text{ and } P_5 = P|_{[4.8,5)}.
\]

Similarly, there are four subprocesses for \( Q \):
\[
Q_1 = Q|_{[0,10)}, \ Q_2 = Q|_{[5,10)}, \ Q_3 = Q|_{[9,10)} \text{ and } Q_4 = Q|_{[9.7,10)}.
\]

5) Define mutually recursive equations:

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

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

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 $\mu$ on $X$ is the supremum of an increasing family of simple measures $\sum_{i=1}^{n} r_i \delta x_i$, where $\sum r_i = 1$.

**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:

- $S$ - states,
- $\mathcal{P}(S) = \{0, 1\}^S$ - predicates over $S$,
- $\text{prog}: \mathcal{P}(S) \to \mathcal{P}(S)$ - predicate transformers - programs.

$\text{prog}(A) = \text{wp}.\text{prog}.A$

Healthiness conditions:

- $\text{prog}(\emptyset) = \emptyset$ - excluded miracle
- $A \subseteq B \Rightarrow \text{prog}(A) \subseteq \text{prog}(B)$ - monotone
- $\text{prog}(A \cap B) = \text{prog}(A) \cap \text{prog}(B)$ - positively conjunctive

Modal operators for $CTL$

- $\text{Next} - \diamond A := \text{prog}(A)$
- $\text{Eventually} - \Diamond A := \mu X. (A \cup \diamond X)$ - least fixed point
- $\text{Always} - \Box A := \nu X. (A \cap \diamond X)$ - greatest fixed point
- $\text{Unless} - A \triangleright B := \nu X. B \cup (A \cap \diamond X)$

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

For example,

$\Box (A \Rightarrow B) \Rightarrow (\Box A \Rightarrow \Box B)$ and $(\Box A \Rightarrow B) \land \diamond A \Rightarrow \diamond B$
How to analyze PCSP?

McIver and Morgan: *Quantitative* temporal logic $q\text{TL}$ for reasoning about *probabilistic processes with demonic nondeterminism* in terms of *expectation transformers*:

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

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

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

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

Kozen, 1983

For nondeterministic $\text{prog}$

$$\text{prog}(E) = \bigcap_{\text{prog} \leq \mu} \int E \text{ d}.\mu$$
How to analyze PCSP?

How to compare expectations:
\( E \supseteq F \) - everywhere no more than
\( E \subseteq F \) - everywhere no less than
\( E \equiv F \) - everywhere equal.

**Healthiness conditions:**

\[
\text{prog}(E) \implies \Box E \quad \text{excluded miracle}
\]
\[
E \supseteq F \implies \text{prog}(E) \supseteq \text{prog}(F) \quad \text{monotone}
\]
\[
\text{prog}(E \not\!
ot\!
ot\not F) = \text{prog}(E) \not\!
ot\!
ot\not \text{prog}(F) \quad \text{subdistributivity}
\]

Modal operators for qTL

Next \(-\quad \Diamond E := \text{prog}(E)\)

Eventually \(-\quad \Diamond E := \mu X.(E \lor \Diamond X) \quad \text{least fixed point}\)

Always \(-\quad \Box E := \nu X.(E \land \Diamond X) \quad \text{greatest fixed point}\)

Unless \(-\quad E \triangleright F := \nu X.F \lor (E \land \Diamond X)\)
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 $0 < r < 1$

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

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

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

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

27
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?