

# Of probabilistic wp and CSP

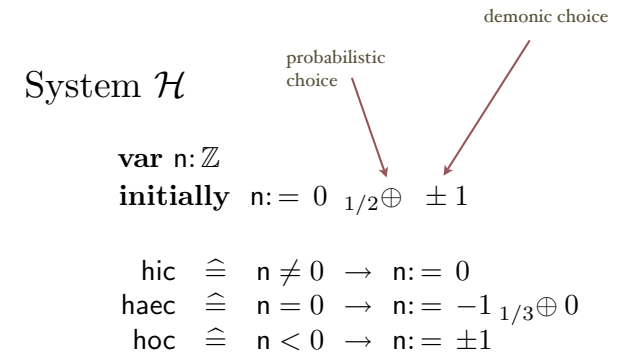
Carroll Morgan  
University of New South Wales

- Action Systems (probabilistic)
- Informal translation to CSP (probabilistic)
- Probabilistic relational model (à la Jifeng He)
- Probabilistic traces via expected values
- Expected-value algebra and duality
- Probabilistic healthiness conditions
- Compositionality?

Carroll Morgan.  
Of wp and CSP.  
In Beauty is our Business, Feijen et al., Springer, 1990.

1

# A probabilistic Action System



R.-J.R. Back and R. Kurki-Suonio.  
Decentralisation of process nets with centralised control.  
2nd ACM SIGACT-SIGOPS Symp. PODC, 131-142, 1983.

2

# Its pCSP “equivalent”

**initially**  $n := 0 \quad 1/2 \oplus \pm 1$

$\text{hic} \hat{=} n \neq 0 \rightarrow n := 0$   
 $\text{haec} \hat{=} n = 0 \rightarrow n := -1 \quad 1/3 \oplus 0$   
 $\text{hoc} \hat{=} n < 0 \rightarrow n := \pm 1$

Process  $\mathcal{H}$

$$\mathcal{H} \hat{=} \mathcal{H}_0 \quad 1/2 \oplus (\mathcal{H}_{-1} \sqcap \mathcal{H}_1)$$

$$\mathcal{H}_1 \hat{=} \text{hic} \rightarrow \mathcal{H}_0$$

$$\mathcal{H}_0 \hat{=} \text{haec} \rightarrow (\mathcal{H}_{-1} \quad 1/3 \oplus \mathcal{H}_0)$$

$$\mathcal{H}_{-1} \hat{=} \text{hic} \rightarrow \mathcal{H}_0 \quad \square \quad \text{hoc} \rightarrow (\mathcal{H}_{-1} \sqcap \mathcal{H}_1)$$

He; Josephs; Woodcock...and M.J. Butler. A CSP Approach to Action Systems.  
DPhil Thesis, 1992.

3

# Its traces

**initially**  $n := 0 \quad 1/2 \oplus \pm 1$

$\text{hic} \hat{=} n \neq 0 \rightarrow n := 0$   
 $\text{haec} \hat{=} n = 0 \rightarrow n := -1 \quad 1/3 \oplus 0$   
 $\text{hoc} \hat{=} n < 0 \rightarrow n := \pm 1$

{  $\langle \rangle$ ,

$\langle \text{hic} \rangle, \langle \text{haec} \rangle, \langle \text{hoc} \rangle,$

$\langle \text{hic}, \text{haec} \rangle,$

$\langle \text{haec}, \text{hic} \rangle,$

$\langle \text{haec}, \text{haec} \rangle,$

$\langle \text{haec}, \text{hoc} \rangle,$

$\langle \text{hoc}, \text{hic} \rangle,$

$\langle \text{hoc}, \text{hoc} \rangle,$

$\langle \text{hic}, \text{haec}, \text{hic} \rangle,$

$\langle \text{hic}, \text{haec}, \text{haec} \rangle,$

$\langle \text{hic}, \text{haec}, \text{hoc} \rangle,$

$\vdots$

}

4

## The relational model for a pAS

5

- A (sub-) distribution  $\Delta$  over a state space  $S$  is a function from  $S$  to  $[0, 1]$  such that  $(\sum_{s:S} \Delta.s) \leq 1$ .
- The set of all such distributions is  $\overline{S}$ .
- A point distribution at  $s$  is written  $\overline{s}$ .
- Non-demonic probabilistic programs have type  $S \rightarrow \overline{S}$ .
- Probabilistic/demonic programs have type  $S \rightarrow \mathbb{P}\overline{S}$ .
- The meaning of a program *prog* is written  $\llbracket prog \rrbracket$ .

Jifeng He, Annabelle McIver and Karen Seidel.  
 Probabilistic models for the guarded command language.  
 Science of Computer Programming 28:171-192, 1997.

## Simple examples

6

- **identity** —  $\llbracket \text{skip} \rrbracket.n = \{\overline{n}\}$   
 The “do-nothing” program **skip** takes any state to itself. Because of our demonic/probabilistic type for programs, however, the result is not just  $n$  again, nor even the set  $\{n\}$ , but rather is the singleton set containing just the point distribution on  $n$ .
- **assignment** —  $\llbracket n := n + 1 \rrbracket.n = \{\overline{n+1}\}$   
 Non-demonic and non-probabilistic assignments deliver singleton sets of point distributions: singleton sets because there is no demonic choice; point-distributions because there is no (non-trivial) probabilistic choice.
- **probabilistic choice** —  $\llbracket n := n + 1 \quad \frac{1}{3} \oplus \quad n := n + 2 \rrbracket.n = \{\Delta'\}$   
 where  $\Delta'.(n+1) = 1/3$   
 $\Delta'.(n+2) = 2/3$   
 $\Delta'.n' = 0$  for other values  $n'$

Non-demonic but probabilistic assignments deliver singleton sets of non-trivial distributions: again the sets are singleton because there is no demonic choice; but the single element of the set is a proper distribution.

1 of 2

## Simple examples

7

- **demonic choice** —  $\llbracket n := n + 1 \quad \sqcap \quad n := n + 2 \rrbracket.n = \{\overline{n+1}, \overline{n+2}\}$   
 A purely demonic (and non-probabilistic) binary choice delivers the distributions contributed by each of its operands.
- **demonic probabilistic choice** —  $\llbracket n := n + 1 \quad \frac{1}{3} \oplus_{1/3} \quad n := n + 2 \rrbracket.n = \{\Delta'_{1/3}, \Delta'_{2/3}\}$   
 where  $\Delta'_p.(n+1) = p$   
 $\Delta'_p.(n+2) = 1-p$   
 $\Delta'_p.n' = 0$  for other values  $n'$

The notation  $_p \oplus_q$ , for  $p+q \leq 1$ , abbreviates the demonic choice between the two probabilistic choices  $_p \oplus$  and  $_{1-q} \oplus$ : it executes the left branch with probability *at least*  $p$ , the right with probability *at least*  $q$  and—in any case—it is certain to execute one or the other.

2 of 2

## Naked guarded commands

8

- A “naked” guarded command (i.e. not “clothed” by an **if...fi**) is executed only if its guard is true; otherwise it “cannot start”.  

$$\Delta' \in \llbracket gd \rightarrow prog \rrbracket.s \iff s \in \llbracket gd \rrbracket \wedge \Delta' \in \llbracket prog \rrbracket.s$$
- If the guard is false in the current state (here  $s$ ), then the result set of distributions is empty.

Miracles of Morris, Nelson, Morgan, late 80's; used e.g. in Event-B and elsewhere.



## Traces of a pAS

13

$$\begin{aligned} \text{hic} &\hat{=} n \geq 0 \rightarrow n := +1 \\ \text{hoc} &\hat{=} n \leq 0 \rightarrow n := -1 \end{aligned}$$

We indicate characteristic functions with square brackets and, in that context, an action name stands for its guard; thus

$$[\text{hic}]_n = \begin{cases} 1 & \text{if } n \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

is the function of the state space  $S$ , that is the integers  $\mathbb{Z}$  here, that is one for non-negative arguments and zero elsewhere.

3 of 6

## Traces of a pAS

14

$$\text{initially } n := -1_{1/2} \oplus +1 \quad \begin{aligned} \text{hic} &\hat{=} n \geq 0 \rightarrow n := +1 \\ \text{hoc} &\hat{=} n \leq 0 \rightarrow n := -1 \end{aligned}$$

The expected value of a characteristic function is the probability assigned to its underlying set. Thus

$$(\text{Exp}.\llbracket ini \rrbracket.\llbracket hic \rrbracket)_n$$

is the probability that event `hic` will be enabled initially.

4 of 6

## Traces of a pAS

15

The probability that `hic` is initially enabled is  $\text{Exp}.\llbracket ini \rrbracket.\llbracket hic \rrbracket_n$

$$\text{initially } n := -1_{1/2} \oplus +1 \quad \begin{aligned} \text{hic} &\hat{=} n \geq 0 \rightarrow n := +1 \\ \text{hoc} &\hat{=} n \leq 0 \rightarrow n := -1 \end{aligned}$$

But  $[\text{hic}]$  itself is the expected value of the constant, everywhere-one function  $[\text{true}]$  over the distributions produced by the action `hic`, provided we take that value to be zero when the guard is false since—in that case—there is no final distribution. Thus we have

$$[\text{hic}]_n = \text{Exp}.\llbracket hic \rrbracket.\llbracket true \rrbracket_n$$

for the characteristic function of `hic`'s guard.

If it's enabled, its probability of establishing true is one; if it's not enabled, its probability of establishing true—or indeed anything at all—is zero.

5 of 6

## Traces of a pAS

16

The probability that `hic` is initially enabled is  $\text{Exp}.\llbracket ini \rrbracket.\llbracket hic \rrbracket_n$

$$\text{initially } n := -1_{1/2} \oplus +1 \quad \begin{aligned} \text{hic} &\hat{=} n \geq 0 \rightarrow n := +1 \\ \text{hoc} &\hat{=} n \leq 0 \rightarrow n := -1 \end{aligned}$$

But  $[\text{hic}]$  itself is the expected value of the constant, everywhere-one function  $[\text{true}]$  over the distributions produced by the action `hic`, provided we take that value to be zero when the guard is false since—in that case—there is no final distribution. Thus we have

$$[\text{hic}]_n = \text{Exp}.\llbracket hic \rrbracket.\llbracket true \rrbracket_n$$

for the characteristic function of `hic`'s guard.

If it's enabled, its probability of establishing true is one; if it's not enabled, its probability of establishing true—or indeed anything at all—is zero.

5 of 6

## Traces of a pAS

17

The probability that **hic** is initially enabled is  $\text{Exp}.\llbracket ini \rrbracket.\llbracket hic \rrbracket.n$

$$\text{initially } n := -1 \text{ } \oplus \text{ } +1 \quad \begin{array}{l} \text{hic} \hat{=} n \geq 0 \rightarrow n := +1 \\ \text{hoc} \hat{=} n \leq 0 \rightarrow n := -1 \end{array}$$

“The probability of the occurrence of the trace  $\langle \text{hic} \rangle$  from initial state  $n$ ”

$$= \text{Exp}.\llbracket ini \rrbracket.\llbracket hic \rrbracket.n$$

$$= \text{Exp}.\llbracket ini \rrbracket.(\text{Exp}.\llbracket hic \rrbracket.\llbracket true \rrbracket).n$$

$$\text{☂} = \text{Exp}.\llbracket ini ; hic \rrbracket.\llbracket true \rrbracket.n .$$

$$\llbracket hic \rrbracket = \text{Exp}.\llbracket hic \rrbracket.\llbracket true \rrbracket$$

6 of

## The algebra of the Exp functions

18

- Because of internal choice (demonic nondeterminism), there may be several final distributions produced by a program *prog*.
- Since standard *CSP* includes a trace if it *could* occur, we take the *maximum* probability over all possible distributions, writing

$$\overline{\text{Exp}}.\llbracket prog \rrbracket.B.s .$$

- $\text{Exp}.\llbracket prog \rrbracket$  and  $\overline{\text{Exp}}.\llbracket prog \rrbracket$  are the same if *prog* is non-demonic.

☂ We have the distribution law

$$\overline{\text{Exp}}.\llbracket prog_1 ; prog_2 \rrbracket.B.s = \overline{\text{Exp}}.\llbracket prog_1 \rrbracket.(\overline{\text{Exp}}.\llbracket prog_2 \rrbracket.B).s .$$

## The Exp duality

19

- Since standard *CSP* includes a trace if it *could* occur, we take the *maximum* probability over all possible distributions, writing

$$\overline{\text{Exp}}.\llbracket prog \rrbracket.B.s .$$

- But if we took the “normal”, demonic view for sequential programs we would take the *minimum* expected value, writing instead

$$\underline{\text{Exp}}.\llbracket prog \rrbracket.B.s .$$

- The two forms are dual, satisfying

$$\overline{\text{Exp}}.\llbracket prog \rrbracket.B.s = 1 - \underline{\text{Exp}}.\llbracket prog \rrbracket.(1 - B).s .$$

☂ And the latter is exactly the greatest pre-expectation for sequential, probabilistic, demonic programs.

1 of 3

## The Exp duality

20

26 1. Introduction to *pGCL*

$$\begin{array}{l} wp.\text{abort}.postE := 0 \\ wp.\text{skip}.postE := postE \\ wp.(x := expr).postE := postE \langle x \mapsto expr \rangle \\ \text{☂ } wp.(prog ; prog').postE := wp.prog.(wp.prog'.postE) \\ wp.(prog \sqcap prog').postE := wp.prog.postE \min wp.prog'.postE \\ wp.(prog \oplus prog').postE := p * wp.prog.postE + \bar{p} * wp.prog'.postE . \end{array}$$

Recall that  $\bar{p}$  is the complement of  $p$ .

The expression on the right gives the greatest pre-expectation of *postE* with respect to each *pGCL* construct, where *postE* is an expression of type  $\mathbb{E}S$  over the variables in state space  $S$ . (For historical reasons we continue to write *wp* instead of *gp*.)

2 of 3

## The Exp duality

21

26 1. Introduction to  $pGCL$

$$\begin{aligned}
 wp.\mathbf{abort}.postE &:= 0 \\
 wp.\mathbf{skip}.postE &:= postE \\
 wp.(x := expr).postE &:= postE \langle x \mapsto expr \rangle \\
 wp.(prog; prog').postE &:= wp.prog.(wp.prog'.postE) \\
 wp.(prog \sqcap prog').postE &:= wp.prog.postE \min wp.prog'.postE \\
 wp.(prog_p \oplus prog').postE &:= p * wp.prog.postE + \bar{p} * wp.prog'.postE
 \end{aligned}$$

Recall that  $\bar{p}$  is the complement of  $p$ .

The expression on the right gives the *greatest pre-expectation* of  $postE$  with respect to each  $pGCL$  construct, where  $postE$  is an expression of type  $\mathbb{E}S$  over the variables in state space  $S$ . (For historical reasons we continue to write  $wp$  instead of  $gp$ .)

C.C. Morgan, A.K. McIver and K. Seidel.  
 Probabilistic predicate transformers.  
 ACM TOPLAS 18(3):325-353, 1996.

2 of 3

## The Exp duality

22

$$\begin{aligned}
 \overline{\text{Exp}}.\llbracket \mathbf{abort} \rrbracket.B &\hat{=} 1 \\
 \overline{\text{Exp}}.\llbracket \mathbf{skip} \rrbracket.B &\hat{=} B \\
 \overline{\text{Exp}}.\llbracket n := expr \rrbracket.B &\hat{=} B_n^{expr} \\
 \overline{\text{Exp}}.\llbracket G \rightarrow prog \rrbracket.B &\hat{=} [G] * \overline{\text{Exp}}.\llbracket prog \rrbracket.B \\
 \overline{\text{Exp}}.\llbracket prog; prog' \rrbracket.B &\hat{=} \overline{\text{Exp}}.\llbracket prog \rrbracket.(\overline{\text{Exp}}.\llbracket prog' \rrbracket.B) \\
 \overline{\text{Exp}}.\llbracket prog \sqcap prog' \rrbracket.B &\hat{=} \max \overline{\text{Exp}}.\llbracket prog \rrbracket.B \\
 &\quad \overline{\text{Exp}}.\llbracket prog' \rrbracket.B \\
 \overline{\text{Exp}}.\llbracket prog_p \oplus prog' \rrbracket.B &\hat{=} p * \overline{\text{Exp}}.\llbracket prog \rrbracket.B \\
 &\quad + (1 - p) * \overline{\text{Exp}}.\llbracket prog' \rrbracket.B
 \end{aligned}$$

26 1. Introduction to  $pGCL$

$$\begin{aligned}
 wp.\mathbf{abort}.postE &:= 0 \\
 wp.\mathbf{skip}.postE &:= postE \\
 wp.(x := expr).postE &:= postE \langle x \mapsto expr \rangle \\
 wp.(prog; prog').postE &:= wp.prog.(wp.prog'.postE) \\
 wp.(prog \sqcap prog').postE &:= wp.prog.postE \min wp.prog'.postE \\
 wp.(prog_p \oplus prog').postE &:= p * wp.prog.postE + \bar{p} * wp.prog'.postE
 \end{aligned}$$

Recall that  $\bar{p}$  is the complement of  $p$ .

The expression on the right gives the *greatest pre-expectation* of  $postE$  with respect to each  $pGCL$  construct, where  $postE$  is an expression of type  $\mathbb{E}S$  over the variables in state space  $S$ . (For historical reasons we continue to write  $wp$  instead of  $gp$ .)

3 of 3

## Probabilistic failures and divergences

23

- The *failures* of a process can be defined similarly: we have that

$$\overline{\text{Exp}}.\llbracket ini; es \rrbracket.\llbracket \neg E \rrbracket.s$$

is the maximum probability that the process can participate in the trace  $es$  and then refuse any event in  $E$ , where  $\llbracket \neg E \rrbracket$  is the characteristic function of the set in which no event of  $E$  is enabled.

- The *divergences* are defined using

$$\overline{\text{Exp}}.\llbracket ini; es \rrbracket.\llbracket \text{false} \rrbracket.s,$$

provided the relational model is (easily) extended to include a bottom “divergent” state  $\perp$ .

- And the *algebra* of  $\overline{\text{Exp}}.\llbracket \cdot \rrbracket$  allows us to prove probabilistic “healthiness laws” about the way these observations interact...

## Standard healthiness

24

$$\begin{aligned}
 \mathbf{C0} & \quad (\langle \rangle, \emptyset) \in F \\
 \mathbf{C1} & \quad (es \uparrow\uparrow es', E) \in F \Rightarrow (es, \emptyset) \in F \\
 \mathbf{C2} & \quad (es, E) \in F \wedge E' \subseteq E \Rightarrow (es, E') \in F \\
 \mathbf{C3} & \quad (es, E) \in F \Rightarrow \begin{aligned} & (es \uparrow\uparrow \langle e \rangle, \emptyset) \in F \\ & \vee (es, E \cup \{e\}) \in F \end{aligned} \\
 \mathbf{C4} & \quad es \in D \Rightarrow es \uparrow\uparrow es' \in D \\
 \mathbf{C5} & \quad es \in D \Rightarrow (es, E) \in F
 \end{aligned}$$

These are the conditions that hold for the failures  $F$  and divergences  $D$  of a standard CSP process.

## Probabilistic healthiness

25

- The probabilistic failure  $\text{pFail}(\text{es}, \mathbf{E})$  is the maximum probability that the action system can perform its initialisation, then execute  $\text{es}$ , then enter a state where no actions in  $\mathbf{E}$  are enabled.

$$\text{pFail}(\text{es}, \mathbf{E}) \hat{=} \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. [\neg \mathbf{E}]$$

- The traces are derived from the failures in the usual way.

$$\text{pTr.es} \hat{=} \text{pFail}(\text{es}, \emptyset)$$

- Divergence is the only way to “establish” false.

$$\text{pDiv.es} \hat{=} \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. [\text{false}]$$

1 of 3

## Probabilistic healthiness

26

To compare these with the standard conditions **C0** to **C5**, read “ $\leq$ ” as implication.

$$\text{pC0} \text{ — } \text{pFail}(\langle \rangle, \emptyset) = 1$$

It is always possible for a system to start, since its initialisation is unguarded.

$$\text{pC1} \text{ — } \text{pFail}(\text{es} ++ \text{es}', \mathbf{E}) \leq \text{pFail}(\text{es}, \emptyset)$$

The probability of continuing a trace is no more than the probability of achieving the trace itself.

$$\text{pC2} \text{ — } \text{pFail}(\text{es}, \mathbf{E}) \geq \text{pFail}(\text{es}, \mathbf{E} \cup \mathbf{E}')$$

The probability of refusing a set of events is no less than the probability of refusing a superset of it.

$$\text{pC3} \text{ — } \text{pFail}(\text{es}, \mathbf{E}) \leq \text{pFail}(\text{es} ++ \langle e \rangle, \emptyset) + \text{pFail}(\text{es}, \mathbf{E} \cup \{e\})$$

If an event cannot be refused, then it must be accepted.

$$\text{pC4} \text{ — } \text{pDiv.es} \leq \text{pDiv}(\text{es} ++ \text{es}')$$

Any event is possible after divergence.

$$\text{pC5} \text{ — } \text{pDiv.es} \leq \text{pFail}(\text{es}, \mathbf{E})$$

Any refusal is possible after divergence.

2 of 3

## Probabilistic healthiness

27

$$\begin{aligned} & \text{pFail}(\text{es} ++ \langle e \rangle, \emptyset) + \text{pFail}(\text{es}, \mathbf{E} \cup \{e\}) \\ \geq & \text{pFail}(\text{es} ++ \langle e \rangle, \emptyset) \parallel \text{pFail}(\text{es}, \mathbf{E} \cup \{e\}) && \text{arithmetic} \\ = & \overline{\text{Exp}}.\llbracket ini; \text{es}; e \rrbracket. [\text{true}] \parallel \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. [\neg(\mathbf{E} \cup \{e\})] && \text{definition pFail} \\ = & \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. [e] \parallel \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. [\neg(\mathbf{E} \cup \{e\})] && \text{sequential composition} \\ \geq & \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. ([e] \parallel [\neg(\mathbf{E} \cup \{e\})]) && \text{super-disjunctivity} \\ \geq & \overline{\text{Exp}}.\llbracket ini; \text{es} \rrbracket. [\neg \mathbf{E}] && \text{set algebra; monotonicity} \\ = & \text{pFail}(\text{es}, \mathbf{E}) && \text{definition pFail} \end{aligned}$$

3 of 3

## The algebra of Exp (reprise)

28

- Sub-conjunctivity (from probabilistic wp):

$$\underline{\text{Exp}}.\llbracket prog \rrbracket.B \ \& \ \underline{\text{Exp}}.\llbracket prog \rrbracket.B' \leq \underline{\text{Exp}}.\llbracket prog \rrbracket.(B \ \& \ B')$$

- Super-disjunctivity (from the above, via duality):

$$\overline{\text{Exp}}.\llbracket prog \rrbracket.B \ \parallel \ \overline{\text{Exp}}.\llbracket prog \rrbracket.B' \geq \overline{\text{Exp}}.\llbracket prog \rrbracket.(B \ \parallel \ B')$$

- Probabilistic conjunction

$$x \ \& \ y \hat{=} (x + y - 1) \ \mathbf{max} \ 0$$

- and its dual, probabilistic disjunction.

$$x \ \parallel \ y \hat{=} (x + y) \ \mathbf{min} \ 1$$

## Conclusions re compositionality

29

- The probabilistic traces, failures and divergences are observations of a probabilistic process, but seem to be too weak to be compositional.
- This is as expected, since the same occurs in sequential programming, but...
- We won't know exactly where the difficulties are until the action-system operations themselves have been defined.
- Of those, parallel composition and hiding seem to be particularly tricky, because of the interaction between probabilistic- and internal choice.

Gavin Lowe.

Representing nondeterministic and probabilistic behaviour in reactive processes.  
[web.comlab.ox.ac.uk/oucl/work/gavin.lowe/papers/prob.html](http://web.comlab.ox.ac.uk/oucl/work/gavin.lowe/papers/prob.html), 1993.

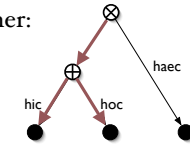
1 of 3

## Conclusions re compositionality

30

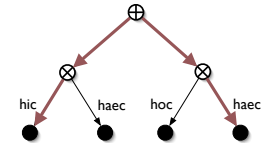
- This process treats hic and hoc without bias, however much it might favour haec over the two together:

$$(hic \rightarrow STOP \text{ }_{1/2} \oplus \text{ }_{1/2} \text{ } hoc \rightarrow STOP) \sqcap haec \rightarrow STOP$$



- But this process might for example execute hic half the time and hoc not at all:

$$_{1/2} \oplus (hic \rightarrow STOP \sqcap haec \rightarrow STOP) \text{ }_{1/2} \oplus (hoc \rightarrow STOP \sqcap haec \rightarrow STOP)$$



- ☹ Yet their pFail observations are identical.

A.K. McIver, C.C. Morgan and J.W. Sanders. Probably Hoare? Hoare Probably!  
 In *Millennial Perspectives in Computer Science*. Davies, Roscoe, Woodcock (eds.), pp 271-282.  
 Palgrave, 2000

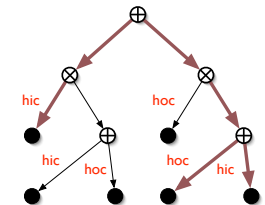
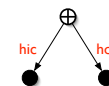
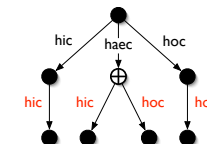
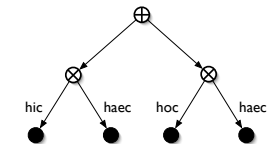
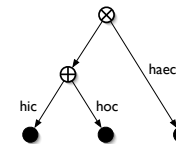
2 of 3

## Conclusions re compositionality

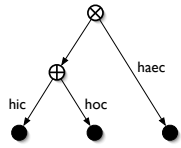
31

- One could extend the pFail and pDiv operations to act on full real-valued expectations rather than only sets of events' guards.
- ☺ This recovers compositionality for sequential programming but, for concurrency, ...
- ?? It might be necessary to extend further, to allow more elaborate "tree-like" and quantitative testing observations, especially for hiding.

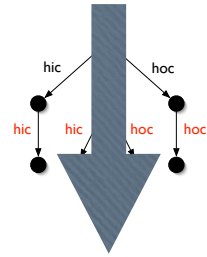
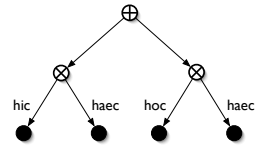
3 of 3



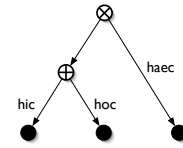
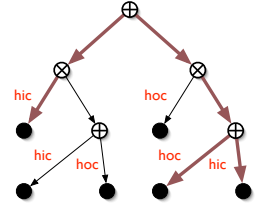
32



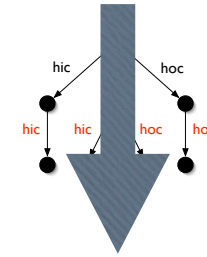
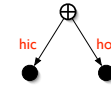
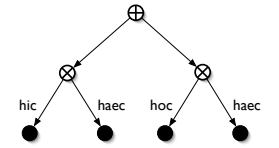
Parallel composition...



...then hide black events.



Parallel composition...



...then hide black events.

