Conditional Entropy and Failed Error Propagation in Software Testing

Is there a problem?

Information theory is a useful level of abstraction at which to model problems in software testing

Intended

input
t1:x==3
t2:x==-5

Unintended

```
x=x+2;
if(x>0)
   x=x%4;
else x=x;
```

output
t1:x==1
t2:x==-3

```
x=3*x;
if(x>0)
   x=x%4;
else x=x;
```

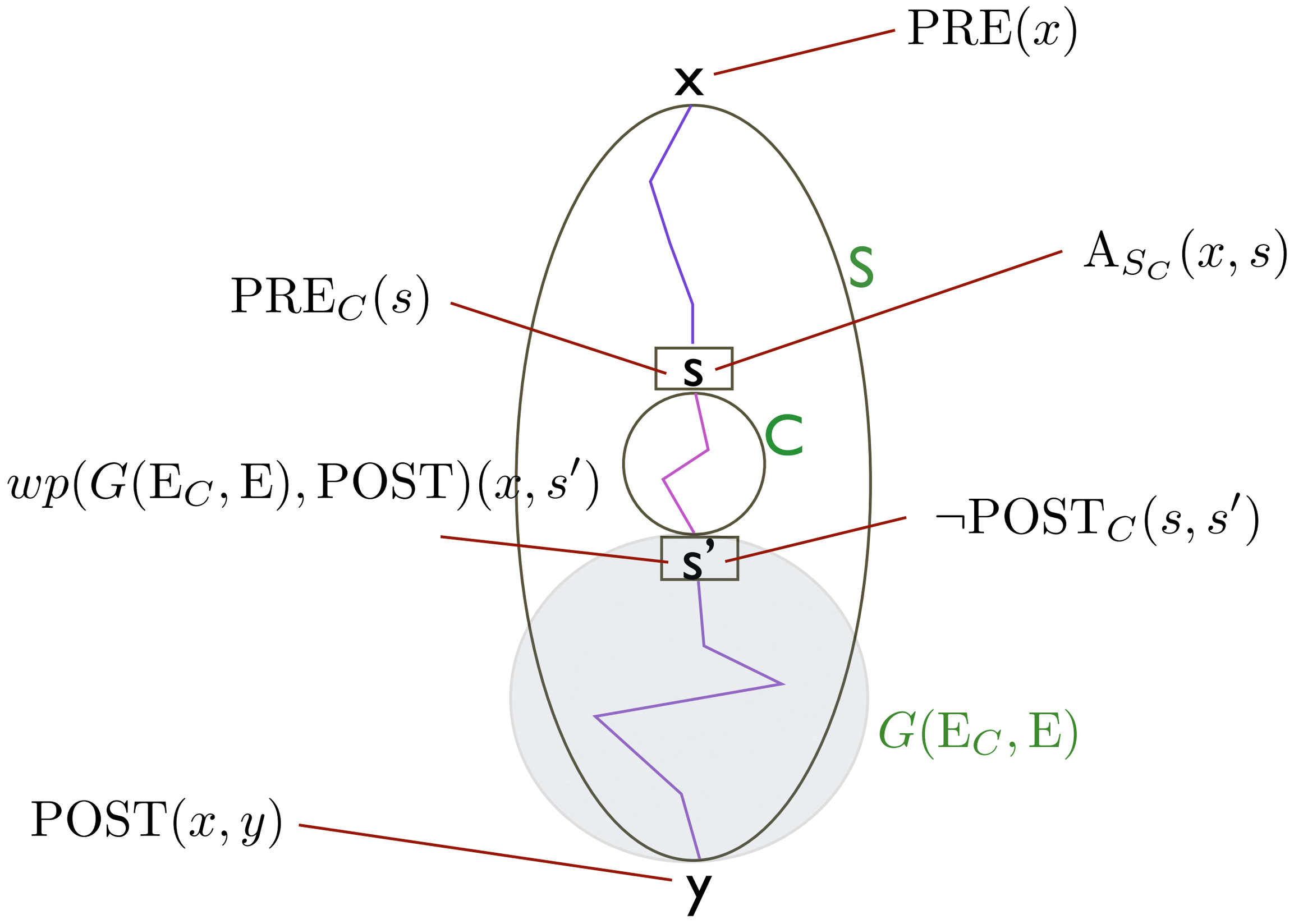
output
t1:x==1
t2:x==-15

software fault masking

- also called **error masking / failed error propagation**
- reduces test set effectiveness
- Error masking condition:

$$\exists x, s, s', y \,.\, \mathrm{PRE}(x) \;\wedge\; \mathrm{A}_{S_C}(x, s) \;\wedge\; \mathrm{PRE}_C(s)$$
$$\wedge\, \neg\mathrm{POST}_C(s, s') \;\wedge\; wp(\mathrm{G}(\mathrm{E}_C, \mathrm{E}), \mathrm{POST})(x, s')$$
$$\wedge\, \mathrm{POST}(x, y)$$

Laski et al. ’95


PRE(x)
x
PRE_C(s)
S
A_{S_C}(x, s)
s
C
wp(G(E_C, E), POST)(x, s')
¬POST_C(s, s')
s'
G(E_C, E)
POST(x, y)
y

- are the sub-programs labelled Q the same?

Not in general, but let us assume that they are.

$$[\![Q]\!]s = [\![Q]\!]s'$$

Domain to Range Ratio

- collisions *necessary*, not *sufficient,* for fault masking
- [Woodward and al-Khanjari (2000)] observed fault masking associated with domain to range ratio
- "loss of information measure" $|D|/|R|$

information theoretic view

Treat the input space and the output space for a program as random variables: I and O

Oracle's Observation of Output

Information in a random variable

$$\mathcal{H}(X) = -\sum_{x \in X} p(x) log_2 p(x)$$

Loss of information from running program **P**

deterministic case

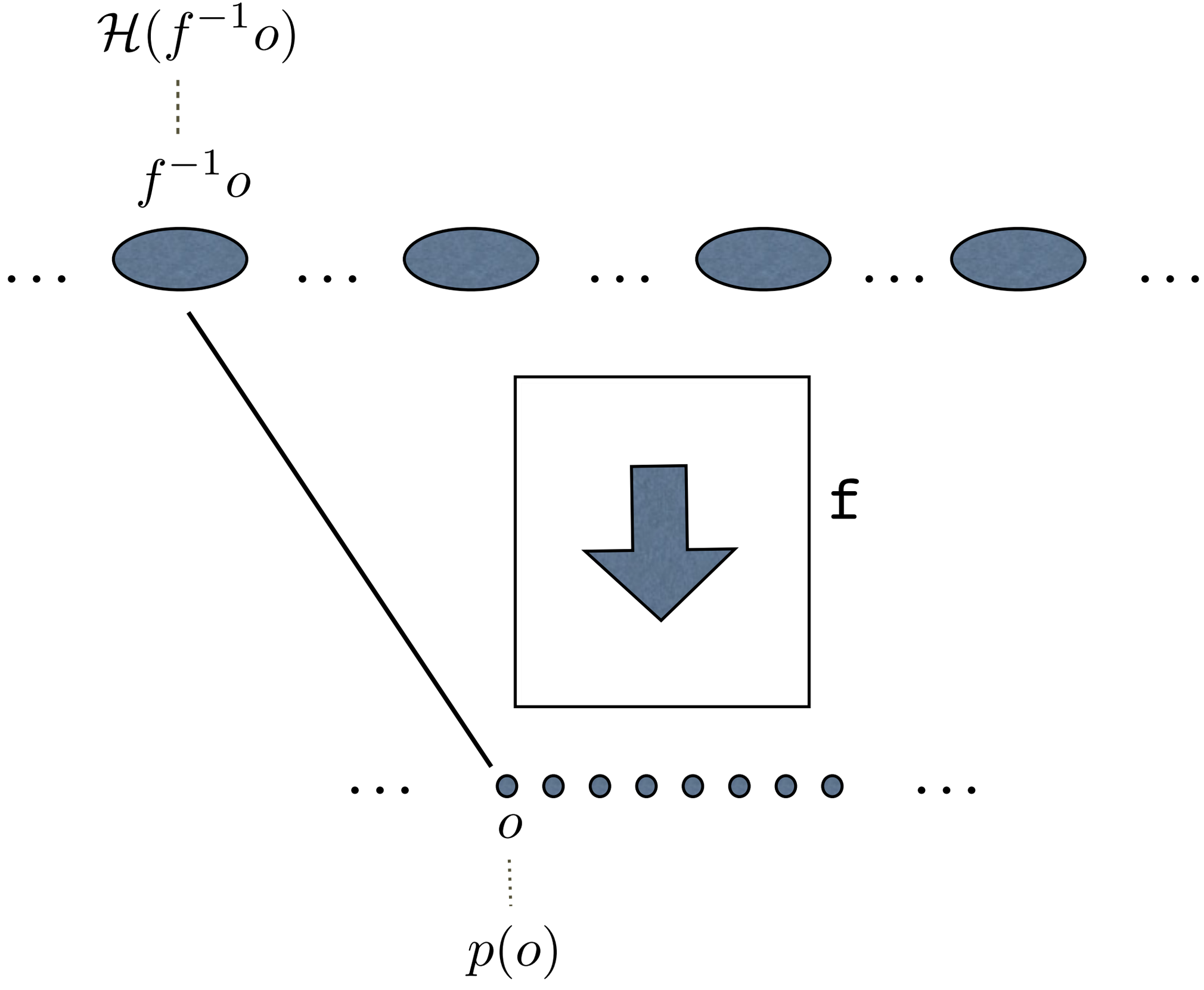
$$\mathcal{H}(I) - \mathcal{H}(O) \quad = \mathcal{H}(I|O)$$

where $[\![P]\!]I = O$

Conditional entropy of I given O: **Squeeziness**.

$$Sq(f) = \mathcal{H}(I) - \mathcal{H}(O) = \sum_{o \in O} p(o)\ \mathcal{H}(f^{-1}o)$$

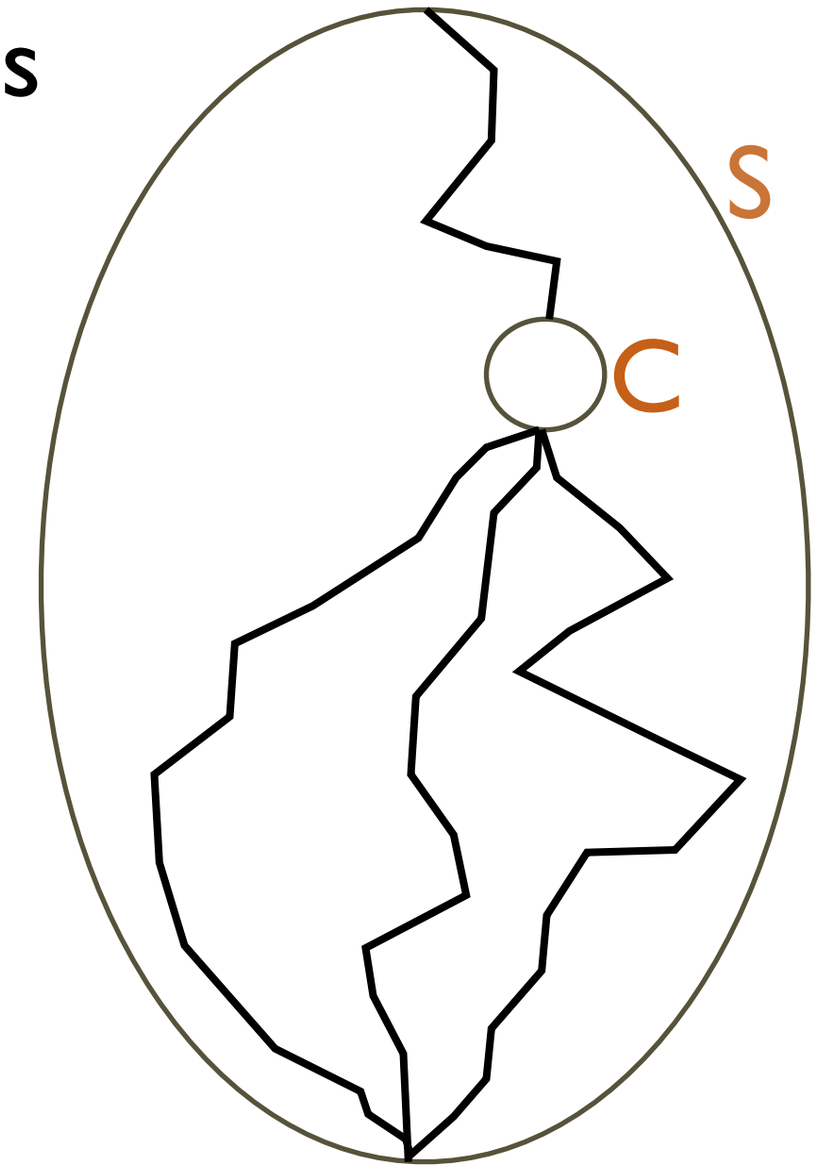
via the partition property


$\mathcal{H}(f^{-1}o)$
$f^{-1}o$
f
o
$p(o)$

what can we do with Squeeziness?

- Measure how much Software Under Test is inclined to fault masking
- Improve test set selection to optimise for individual test effectiveness

Use covering paths to generate tests



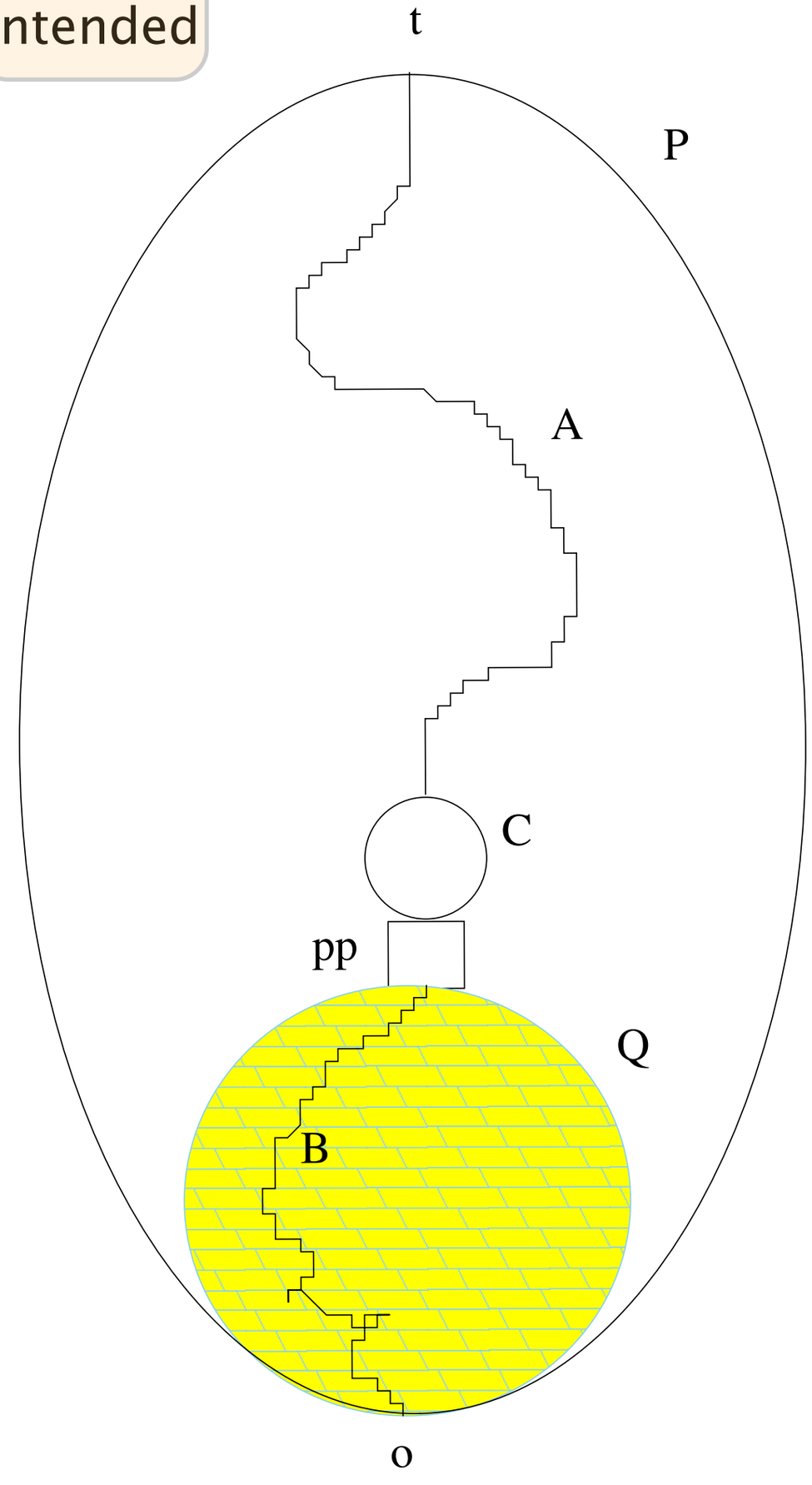
Pick a “less Squeezy" path

Reduce possible fault masking

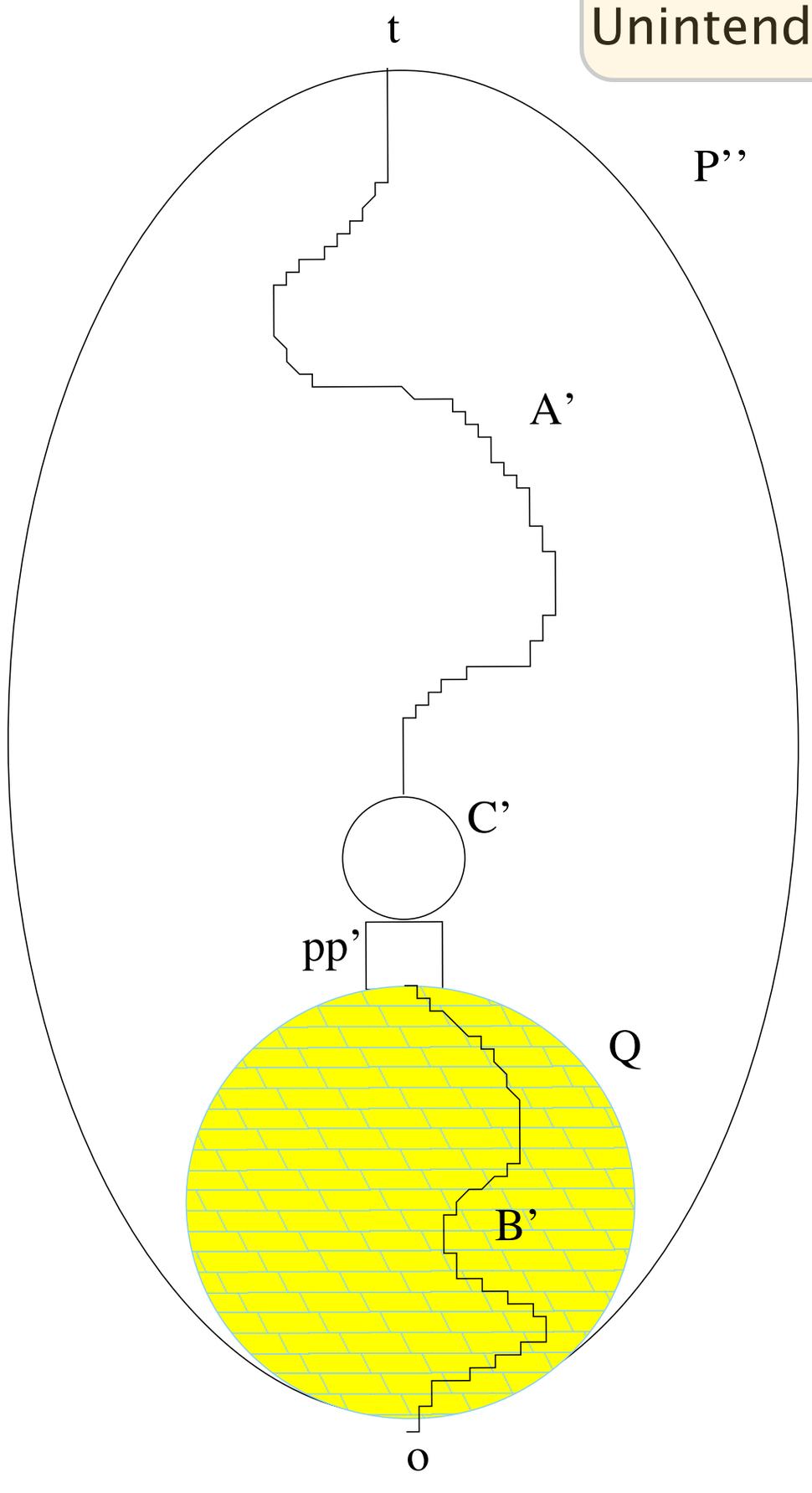
experimental validation

- consider statement coverage
- examine correlation between the probability of failed error propagation and squeeziness for different parts of a program
- use mutation testing setting so we have both the *intended* and the *unintended* program (= original and mutant)

Intended



Unintended



assumptions

- single error in each program (mutant)
- mutation of assignment statements only
- Q is the same in both programs
- non-induced probability distributions are uniform p.d. (MEP)

how easy is it to find an effective test input that covers a given statement?

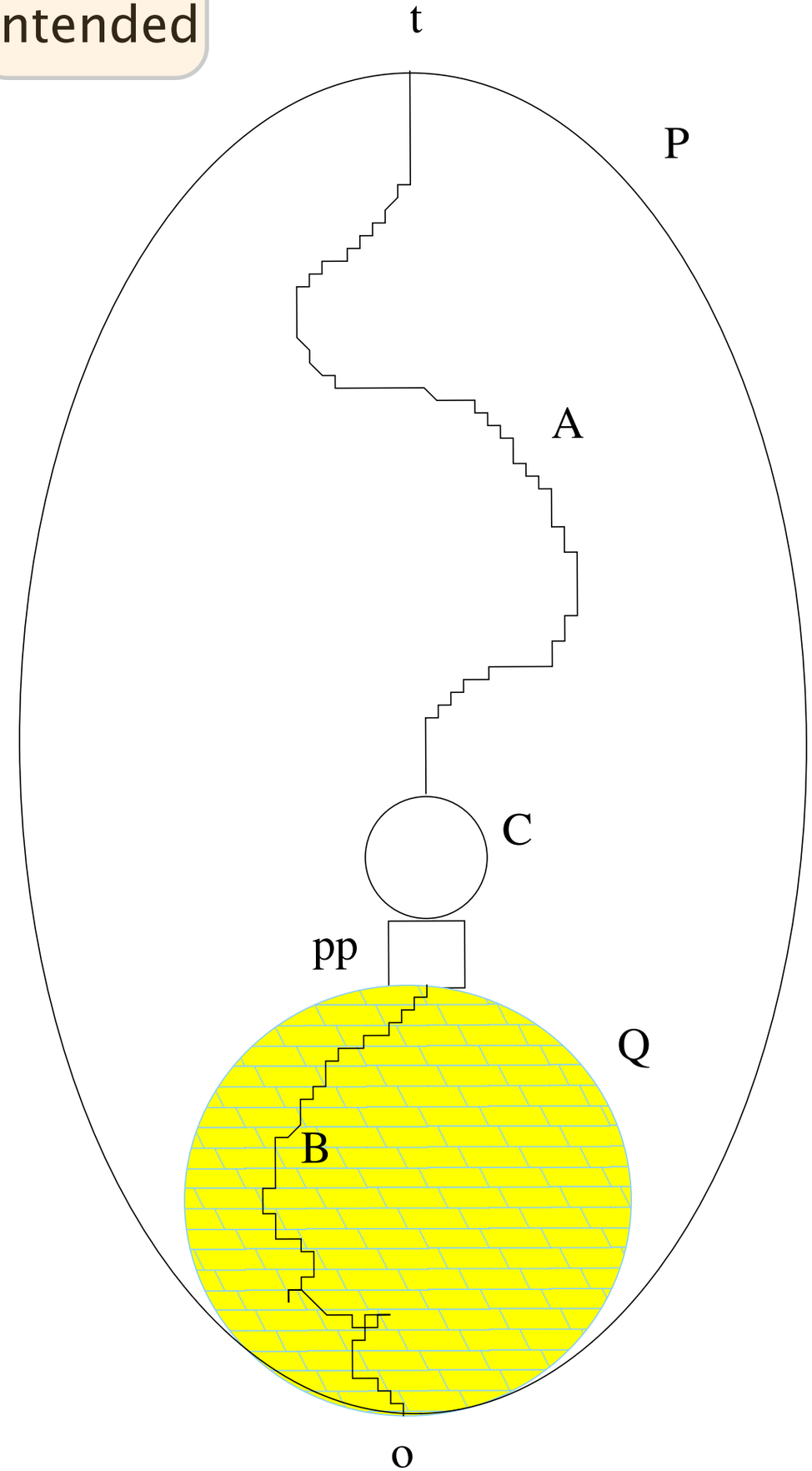
answer this by examining the information flow behaviour of Q

Hypothesis 1

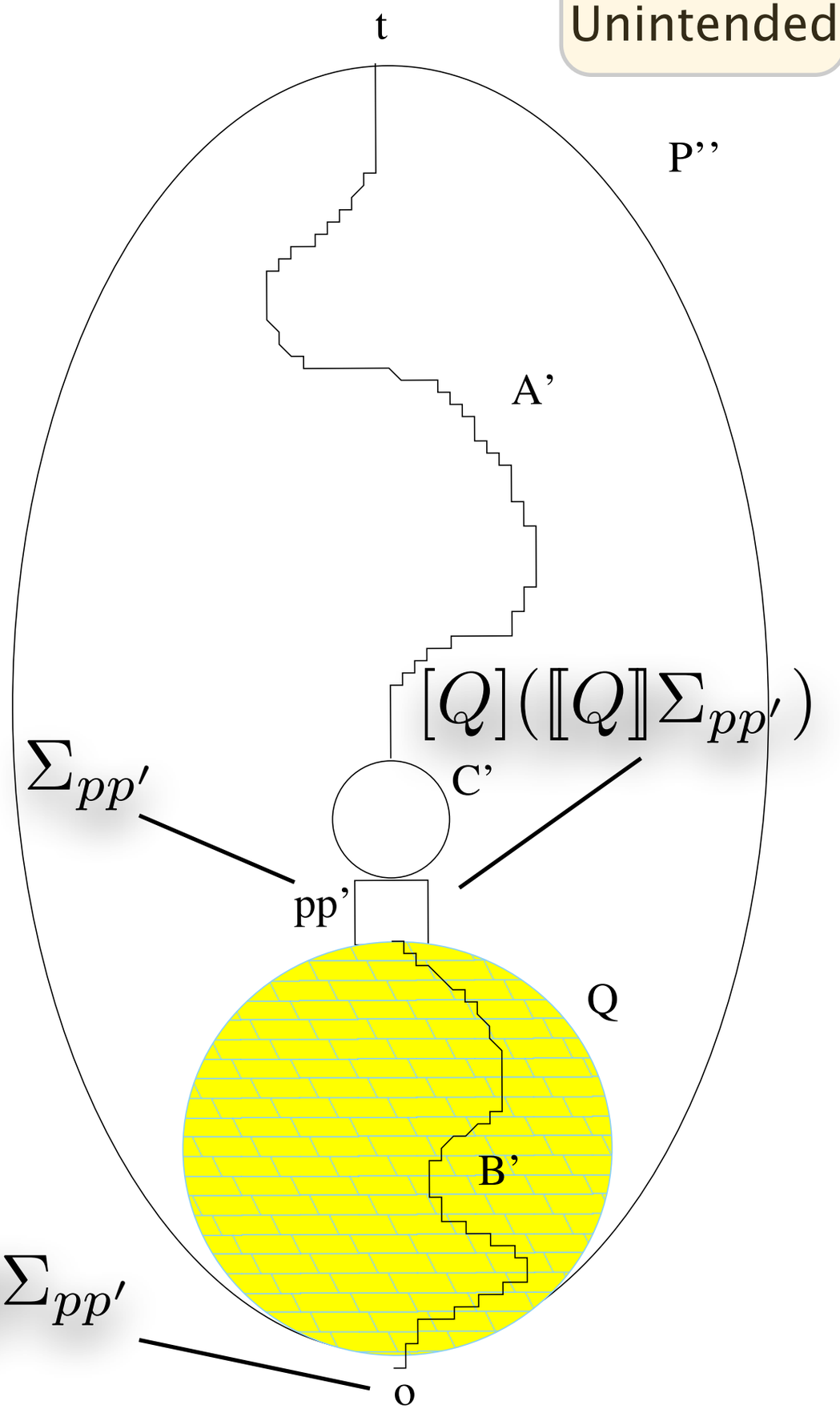
There is a correlation between
the probability of FEP for all input states
whose execution path includes pp' and

$$sq([\![Q]\!], [Q]([\![Q]\!]\Sigma_{pp'}))$$

Intended



Unintended



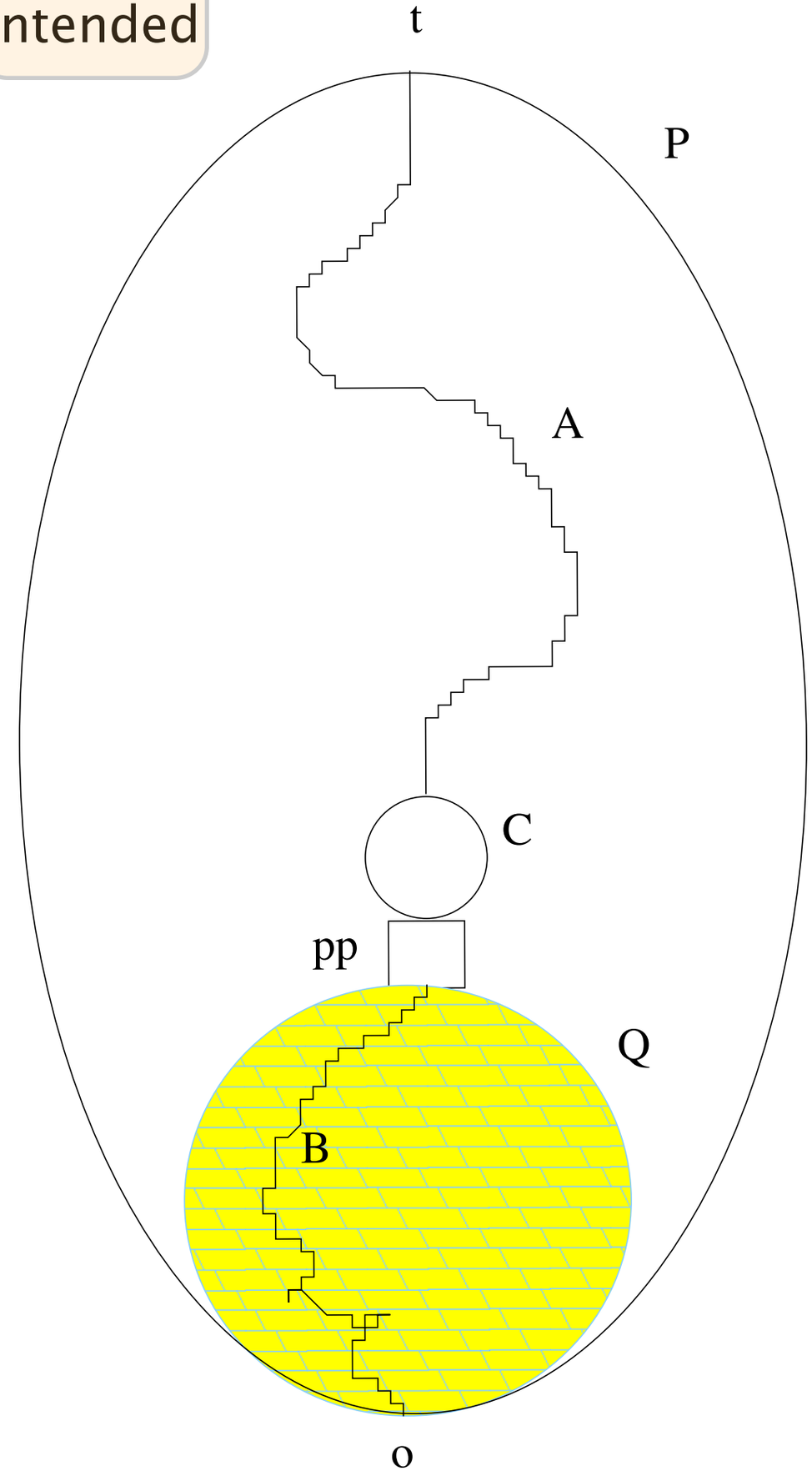
Hypothesis 2

There is a correlation between the probability of FEP for all input states that reach pp' via the execution of R' and

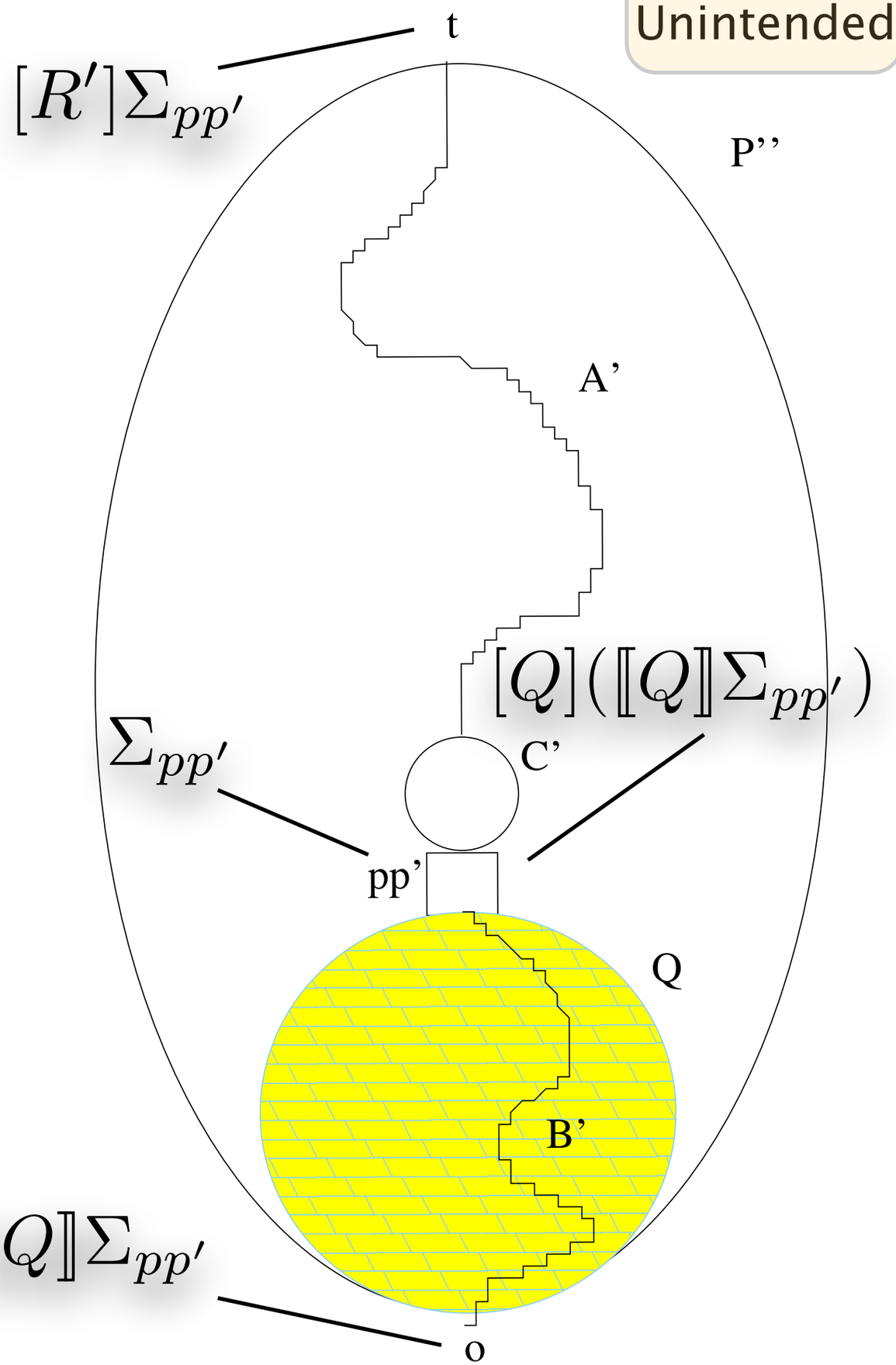
$$sq([\![R']\!], [R']\Sigma_{pp'}) + sq([\![Q']\!], [Q']([\![Q']\!]\Sigma_{pp'}))$$

R' is the sub program of P' that is backwardly reachable from pp'

Intended



Unintended



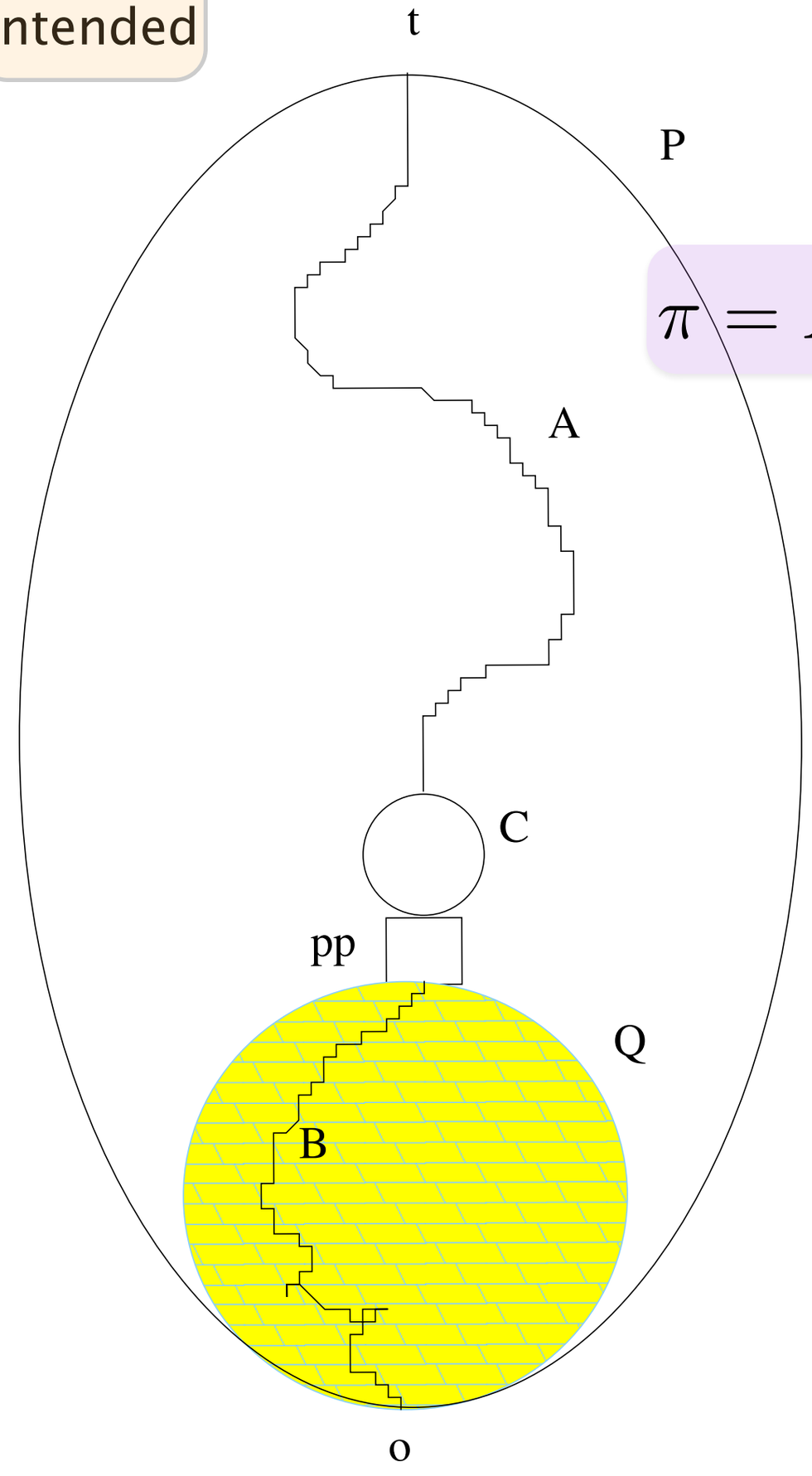
Hypothesis 3

There is a correlation between the probability of FEP for all states that reach pp' via execution along a path and

$[\![\pi]\!]^{pa}_{pp'}$

$$sq([\![Q']\!], [Q']([\![\pi]\!]\Sigma_I))$$

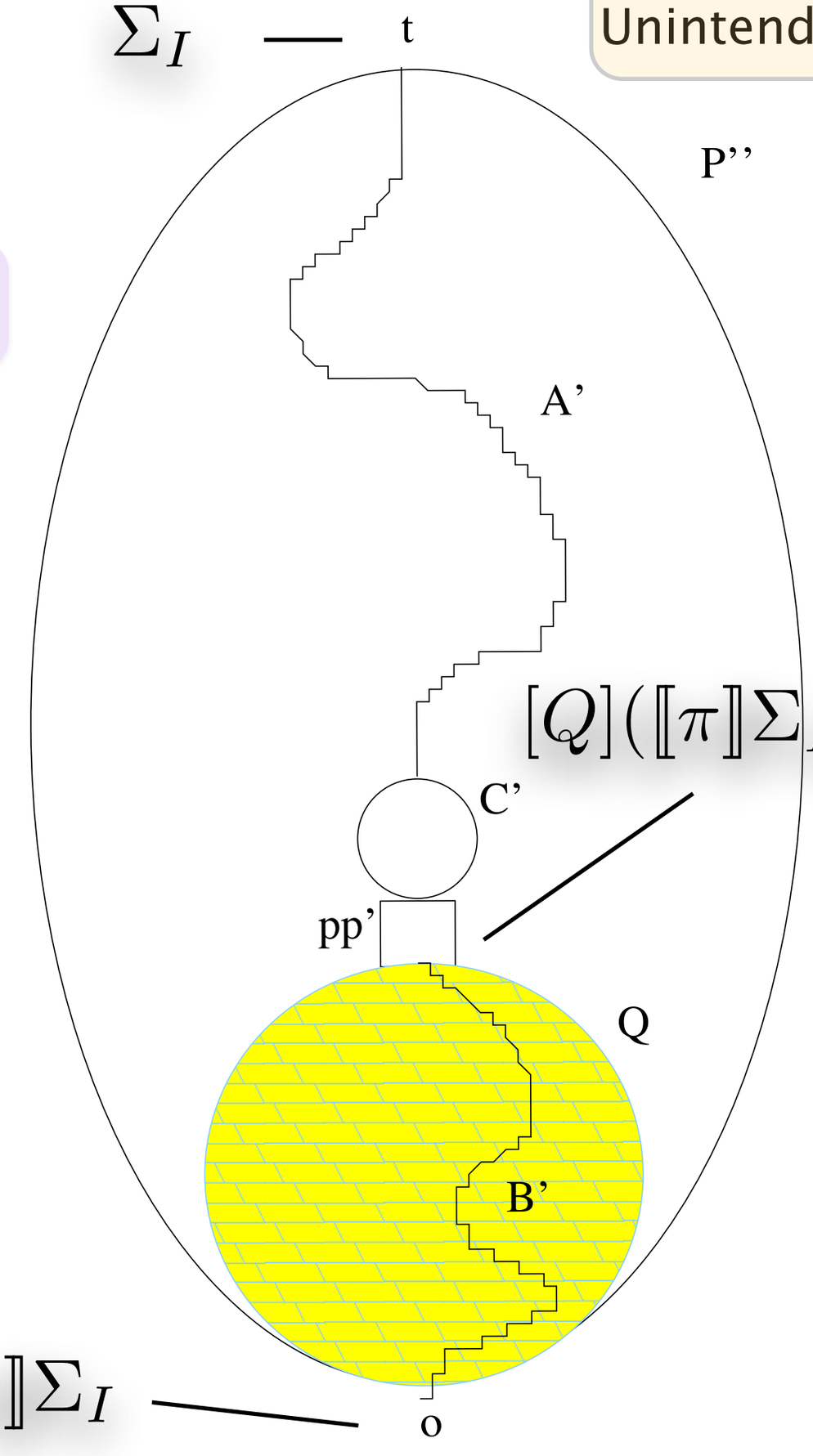
Intended



$\pi = A'B'$

Σ_I

Unintended



$[Q](\llbracket\pi\rrbracket\Sigma_I)$

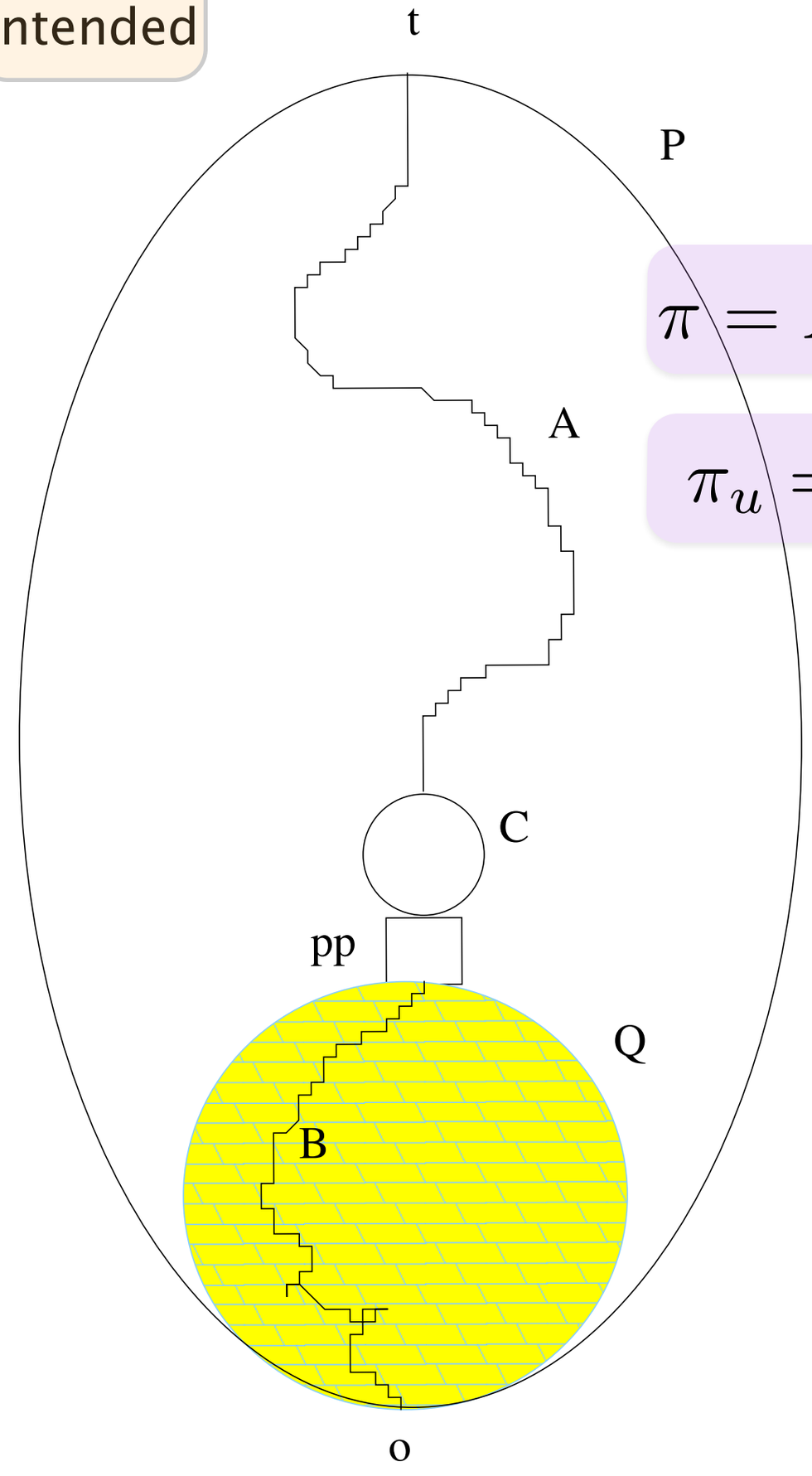
$\llbracket\pi\rrbracket\Sigma_I$

Hypothesis 4

There is a correlation between the probability of FEP for all states that reach pp' via execution along a path and

$$sq([\![\pi_u]\!], [\pi_u][\![\pi]\!]^{pa}_{pp'}) + sq([\![Q']\!], [Q']([\![\pi]\!]\Sigma_I))$$

Intended

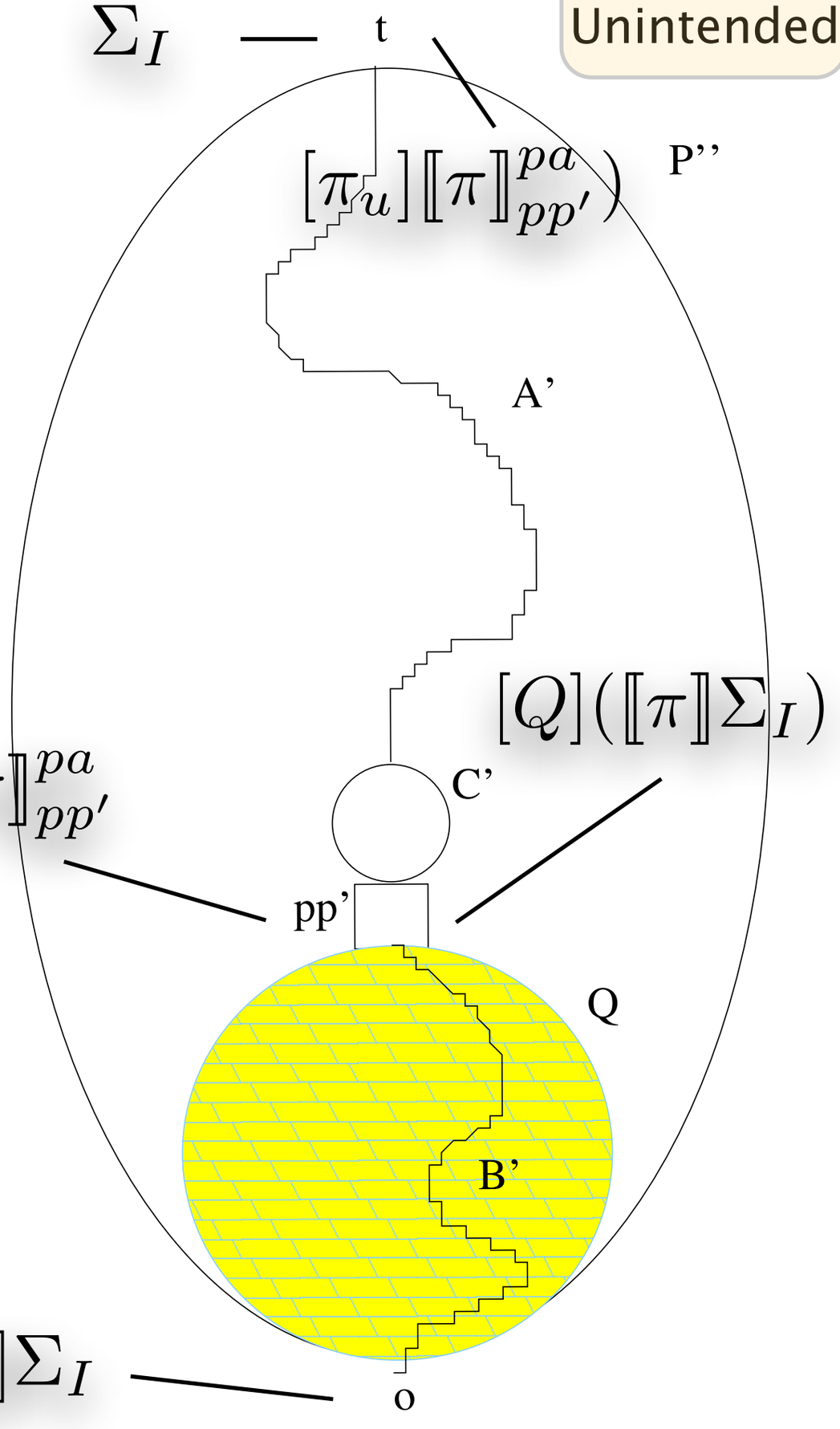


$\pi = A'B'$

$\pi_u = A'$

Σ_I

Unintended



$[\pi_u][\![\pi]\!]^{pa}_{pp'}$

$[Q]([\![\pi]\!]\Sigma_I)$

$[\![\pi]\!]^{pa}_{pp'}$

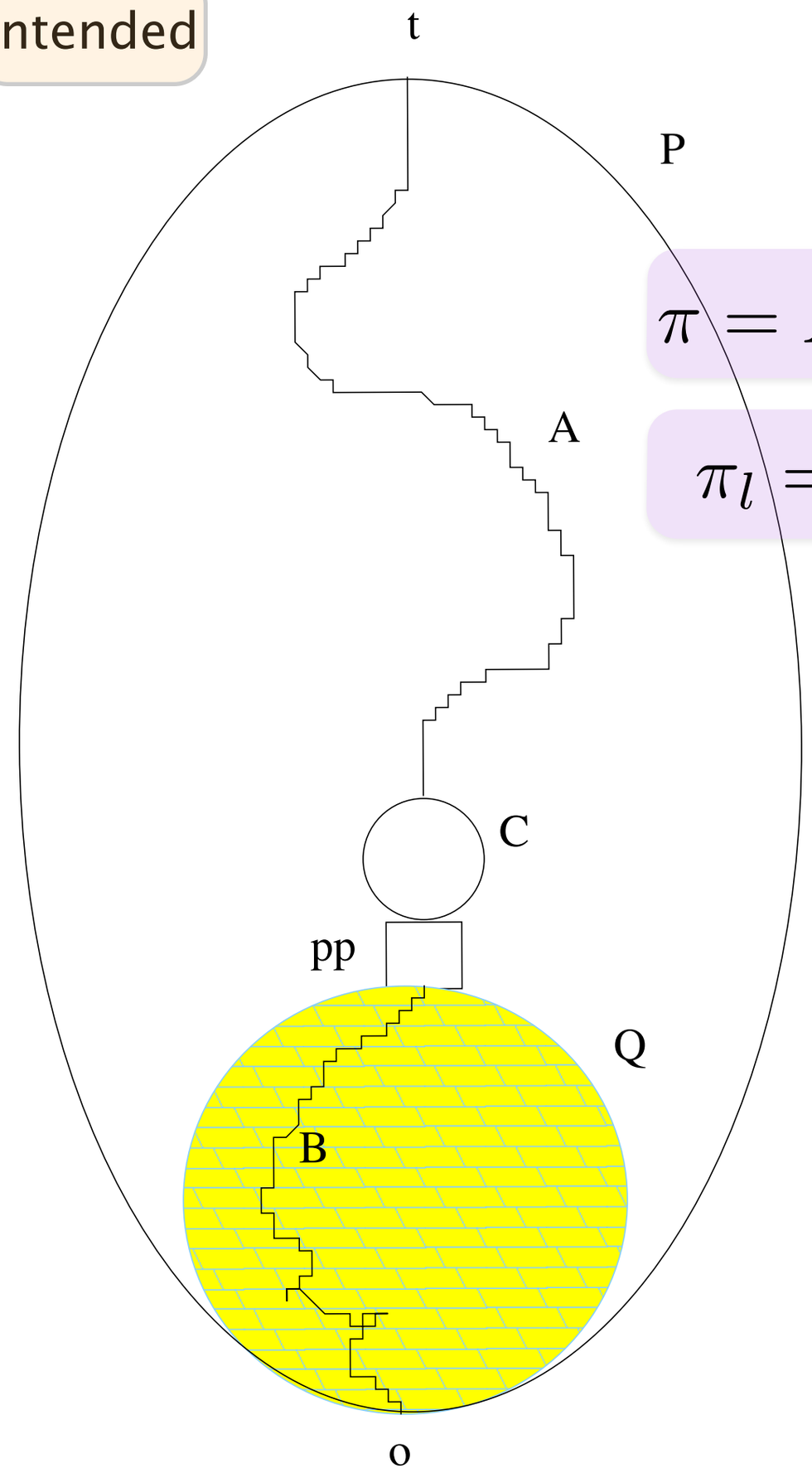
$[\![\pi]\!]\Sigma_I$

Hypothesis 5

There is a correlation between the probability of FEP for all states that reach pp' via execution along a path and

$$sq([\![\pi_l]\!], [\![\pi]\!]^{pa}_{pp'}))$$

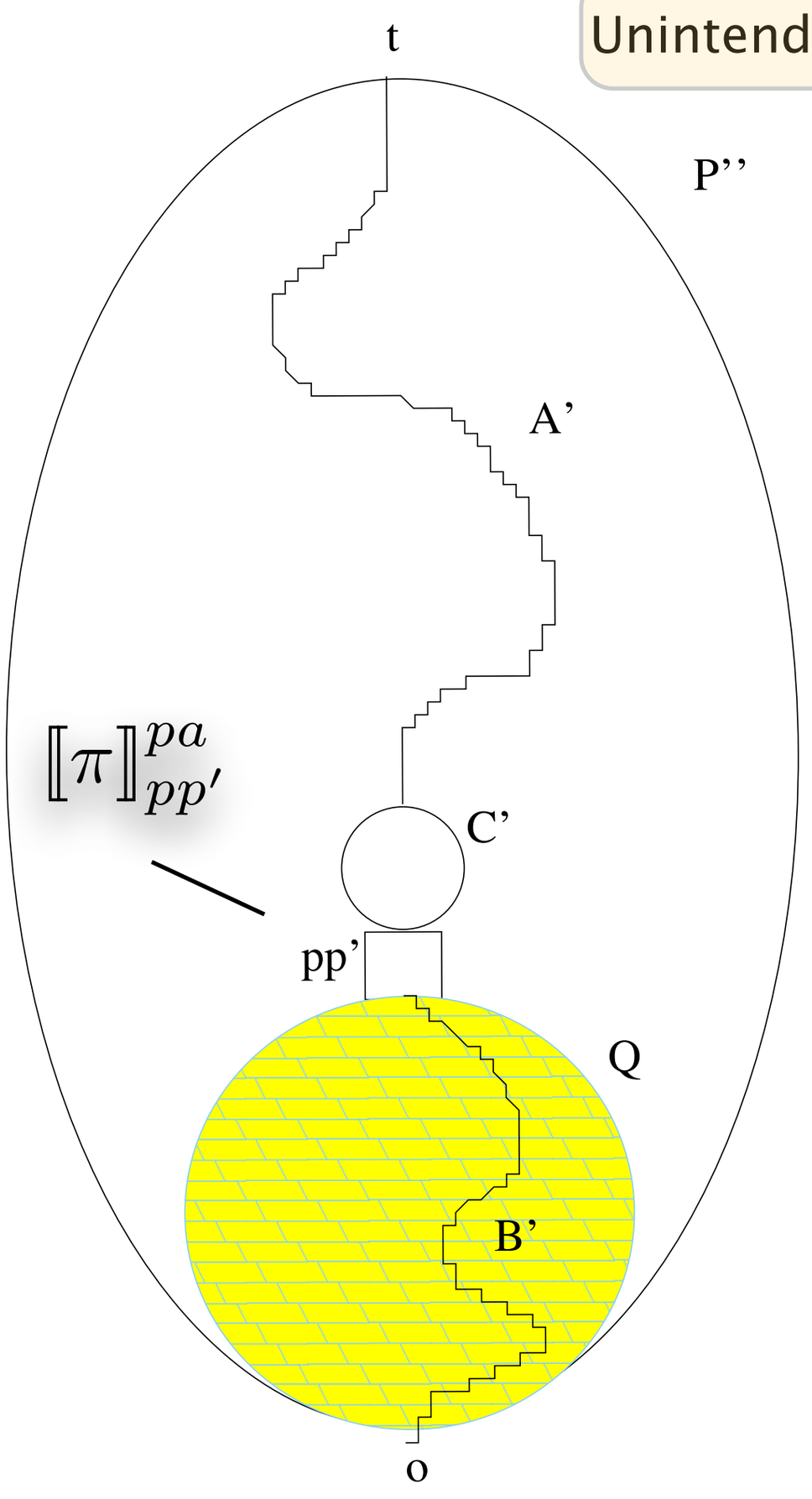
Intended



$$\pi = A'B'$$

$$\pi_l = B'$$

Unintended



$$[\![\pi]\!]^{pa}_{pp'}$$

a squeeziness close to zero for a path means
that we don't need to rank the covering paths
but can simply use that path to generate
a test input to cover the program construct.

Hypothesis 6

$\epsilon > 0$ **small,**

$$sq([\![Q']\!], [Q'](\![\pi]\!]\Sigma_I)) \leq \epsilon \Longrightarrow p(FEP) \leq \epsilon$$

- 30 programs
- 7,140,000 test cases
- five metrics
- two metrics have 0.95 Spearman rank correlation with p(FEP)
- 10% of test cases suffer from p(FEP)

Table 1: Projects under investigation

Project	Function	total LoC	Mutants
Toy	17	810	383
R	10	221k	953
GRETL	3	286k	72

- Seeded faults into each program
- C mutation operator tool OAAN
- Generate mutants with SMT-C
- Gnu Debugger to extract internal states

Table 2: Real world statistical subject programs

Project	Function	C Files	LoC	SLoC
R	bratio	4	1667	1573
	rhyper	2	338	260
	gamma_cody	2	137	116
	ptukey	7	1360	674
	qgamma	14	2397	1312
	psi	2	261	119
	pnorm_both	1	315	178
	pnchisq_raw	1	275	181
	gammafn	5	527	264
	qt	1	234	124
GRETL	i0	2	270	108
	k0	5	688	267
	unity	3	335	147

$$\text{p(FEP)} = \frac{\begin{array}{c}\#\text{ of tests that weakly kill } P' \\ \text{but do not strongly kill } P'\end{array}}{\#\text{ of tests that weakly kill } P'}$$

EXP1 and EXP2: number of inputs that reach pp' via any path

EXP3, EXP4 and EXP5: number of inputs that reach pp' via a single execution path

$$[Q](\llbracket Q \rrbracket \Sigma_{pp'}) \longrightarrow \Sigma_{pp'} \cup \Sigma_{pp}$$

$$[Q](\llbracket \pi \rrbracket \Sigma_I) \longrightarrow \llbracket \pi \rrbracket^{pa}_{pp'} \cup \llbracket \pi \rrbracket^{pa}_{pp}$$

Table 3: The proportion of randomly generated tests for all subject programs that are weakly and strongly killed.

	Weakly Killed	Strongly Killed	Proportion
EP	Yes	Yes	84.73 %
FEP	Yes	No	9.85 %
CC1	No	No	4.89 %
CC2	No	Yes	0.44 %

Table 4: Spearman's Rank Correlation Coefficient for all programs.

Experiment	Correlation
EXP1	0.715267
EXP2	0.699165
EXP3	0.955647
EXP4	0.948299
EXP5	0.031510

Table 7: Maximum p(FEP) for all programs

sq(Q') Range	Max sq(Q)	Max p(FEP)
≤ 0.1	0.090683	0.090683
≤ 0.01	0.001120	0.001120
≤ 0.001	0.000800	0.000200