Noah Goodman

Concepts in a Probabilistic Language of Thought

Journal Club

at Wigner

Probabilistic language of thought hypothesis
(informal version)

symbolic approach statistical approach

Probabilistic language of thought hypothesis (informal version)

symbolic approach

statistical approach

compositionality

Probabilistic language of thought hypothesis (informal version)

symbolic approach

compositionality

statistical approach

gradedness / uncertainty

Probabilistic language of thought hypothesis (informal version)

symbolic approach

compositionality

statistical approach

gradedness / uncertainty

Probabilistic language of thought hypothesis (informal version)

generative process / simulation

logic

atomic propositions (variables)

cough	sneeze	flu	TB

logic

atomic propositions (variables)

cough	sneeze	flu	TB	possible?
t	t	t	t	y

defining a possible world

logic

cough	sneeze	flu	TB	possible?
t	t	t	t	y
t	t	f	f	n
...	...	...	...	...
f	f	f	f	y

inference

inference

$$cough, \neg flu \vdash TB?$$

inference

$$cough, \neg flu \vdash TB?$$

cough	flu	TB	possible?
t	t	t	y
t	t	f	y
t	f	t	y
t	f	f	n
f	t	t	n
f	t	f	n
f	f	t	n
f	f	f	y

inference

$$cough, \neg flu \vdash TB?$$

cough	flu	TB	possible?
~~t~~	~~t~~	~~t~~	~~y~~
t	t	f	y
t	f	t	y
t	f	f	n
f	t	t	n
f	t	f	n
f	f	t	n
f	f	f	y

inference

$$cough, \neg flu \vdash TB?$$

cough	flu	TB	possible?
~~t~~	~~t~~	~~t~~	~~y~~
~~t~~	~~t~~	~~f~~	~~y~~
t	f	t	y
t	f	f	n
f	t	t	n
f	t	f	n
f	f	t	n
f	f	f	y

} has flu

inference

$$cough, \neg flu \vdash TB?$$

cough	flu	TB	possible?
~~t~~	~~t~~	~~t~~	~~y~~
~~t~~	~~t~~	~~f~~	~~y~~
t	f	t	y
t	f	f	n
~~f~~	~~t~~	~~t~~	~~n~~
f	t	f	n
f	f	t	n
f	f	f	y

} has flu

inference

$$cough, \neg flu \vdash TB?$$

cough	flu	TB	possible?
~~t~~	~~t~~	~~t~~	~~y~~
~~t~~	~~t~~	~~f~~	~~y~~
t	f	t	y
t	f	f	n
~~f~~	~~t~~	~~t~~	~~n~~
~~f~~	~~t~~	~~f~~	~~n~~
~~f~~	~~f~~	~~t~~	~~n~~
~~f~~	~~f~~	~~f~~	~~y~~

has flu

no cough

inference

$$cough, \neg flu \vdash TB?$$

cough	flu	TB	possible?
t	t	t	y
t	t	f	y
t	f	t	y
t	f	f	n
f	t	t	n
f	t	f	n
f	f	t	n
f	f	f	y

has flu

no cough

inference

$cough, \neg flu \vdash TB?$ **true**

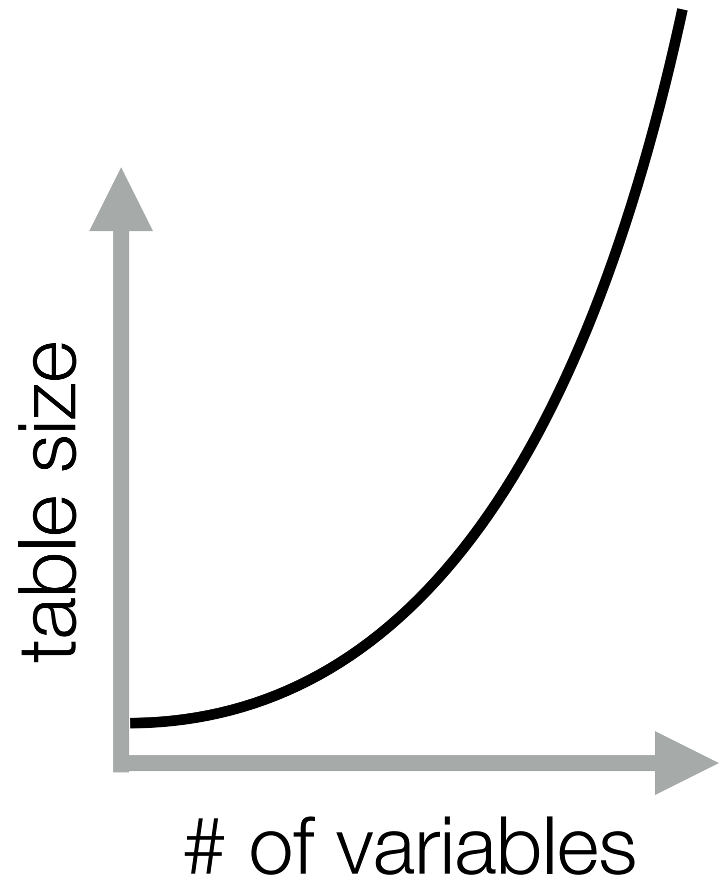
cough	flu	TB	possible?
~~t~~	~~t~~	~~t~~	~~y~~
~~t~~	~~t~~	~~f~~	~~y~~
t	f	t	y
t	f	f	n
~~f~~	~~t~~	~~t~~	~~n~~
~~f~~	~~t~~	~~f~~	~~n~~
~~f~~	~~f~~	~~t~~	~~n~~
~~f~~	~~f~~	~~f~~	~~y~~

has flu

no cough

problems?

problems?



solution

propositional logic
introduce operators to describe possible worlds more compactly by composition

A	B	$A \vee B$
t	t	t
t	f	t
f	t	t
f	f	f

solution

cough	flu	TB	possible?
t	t	t	y
t	t	f	y
t	f	t	y
t	f	f	n
f	t	t	n
f	t	f	n
f	f	t	n
f	f	f	y

⟷

$$(flu \vee TB) \leftrightarrow cough$$

higher-order logics

truth table

higher-order logics

truth table

propositional logic

expressive power

higher-order logics

truth table

boolean operators:

truth val. to truth val.

propositional logic

expressive power

higher-order logics

truth table

boolean operators:
truth val. to truth val.

propositional logic

first-order logic

higher-order logics

boolean operators:
truth val. to truth val.

predicates:
objects to truth val.

truth table

propositional logic

first-order logic

expressive power

higher-order logics

truth table

boolean operators:
truth val. to truth val.

propositional logic

predicates:
objects to truth val.

first-order logic

quantifiers:
object var. to object

expressive power

higher-order logics

truth table

boolean operators:
truth val. to truth val.

propositional logic

predicates:
objects to truth val.

first-order logic

quantifiers:
object var. to object

rules of chess:
propositional logic - 1000 p
first order logic - 1p

expressive power

higher-order logics

truth table

boolean operators:
truth val. to truth val.

propositional logic

predicates:
objects to truth val.

first-order logic

quantifiers:
object var. to object

universality

expressive power

higher-order logics

truth table

boolean operators:
truth val. to truth val.

propositional logic

predicates:
objects to truth val.

first-order logic

quantifiers:
object var. to object

universality

λ-calculus

expressive power

higher-order logics

truth table

boolean operators:
truth val. to truth val.

propositional logic

predicates:
objects to truth val.

first-order logic

quantifiers:
object var. to object

universality

λ-calculus

(or UTM, partial recursive f., second-order logic, etc.)

expressive power

logic

cough	sneeze	flu	TB	possible?
t	t	t	t	y
t	t	f	f	n
...	...	...	...	...
f	f	f	f	y

probability

cough	sneeze	flu	TB	probability
t	t	t	t	0.1
t	t	f	f	0.02
...	...	...	...	...
f	f	f	f	0.3

inference

cough	sneeze	flu	TB	probability
t	t	t	t	0.1
t	t	f	f	0.02
...	...	...	...	...
f	f	f	f	0.3

inference **is conditioning**

cough	sneeze	flu	TB	probability
t	t	t	t	0.1
t	t	f	f	0.02
...	...	...	...	...
f	f	f	f	0.3

inference **is conditioning**

$$P(TB|cough, \neg flu) = ?$$

cough	sneeze	flu	TB	probability
t	t	t	t	0.1
t	t	f	f	0.02
...	...	...	...	...
f	f	f	f	0.3

inference **is conditioning**

$$P(TB|cough, \neg flu) = ?$$

cough	sneeze	flu	TB	probability
~~t~~	~~t~~	~~t~~	~~t~~	~~0.1~~
t	t	f	f	0.02
...	...	...	...	...
f	f	f	f	0.3

has flu

inference **is conditioning**

$$P(TB|cough, \neg flu) = ?$$

cough	sneeze	flu	TB	probability
~~t~~	~~t~~	~~t~~	~~t~~	~~0.1~~
t	t	f	f	0.02
...	...	...	...	...
~~f~~	~~f~~	~~f~~	~~f~~	~~0.3~~

has flu

no cough

inference **is conditioning**

$$P(TB|cough, \neg flu) = ?$$

cough	sneeze	flu	TB	probability
~~t~~	~~t~~	~~t~~	~~t~~	~~0.1~~
t	t	f	f	0.02
...	...	...	...	...
~~f~~	~~f~~	~~f~~	~~f~~	~~0.3~~

has flu

no cough

inference **is conditioning**

$$P(TB|cough, \neg flu) = ?$$

cough	sneeze	flu	TB	probability
t	t	t	t	0.1
t	t	f	f	0.02
...	...	...	...	...
f	f	f	f	0.3

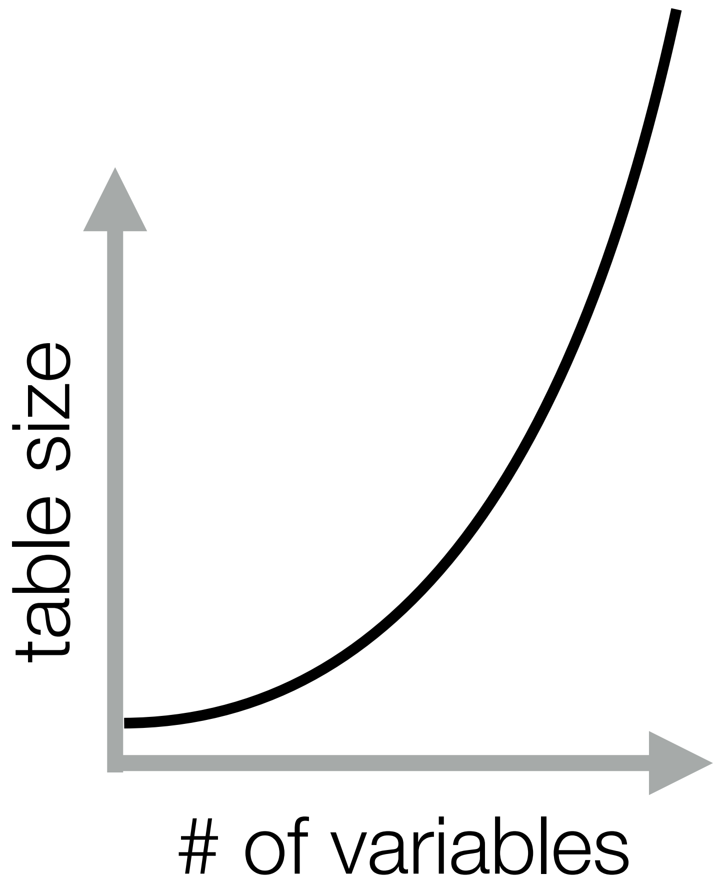
has flu

no cough

renormalise what remains

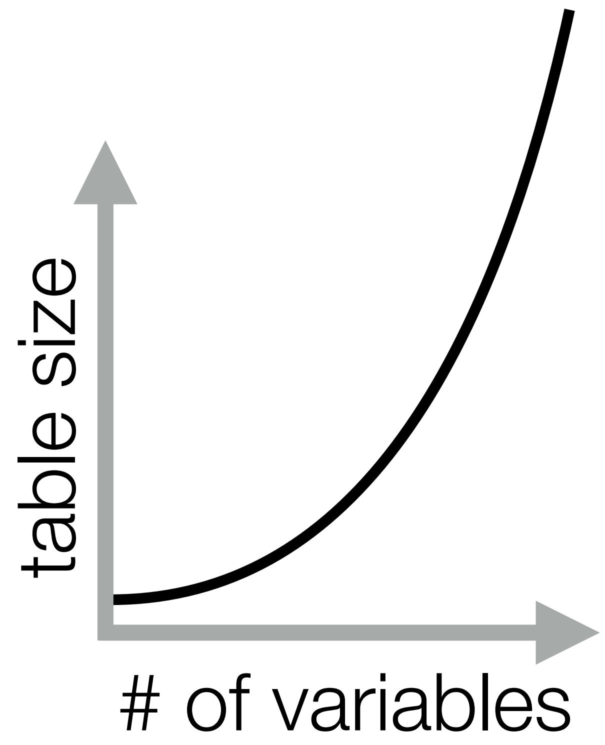
problems?

problems?



problems?

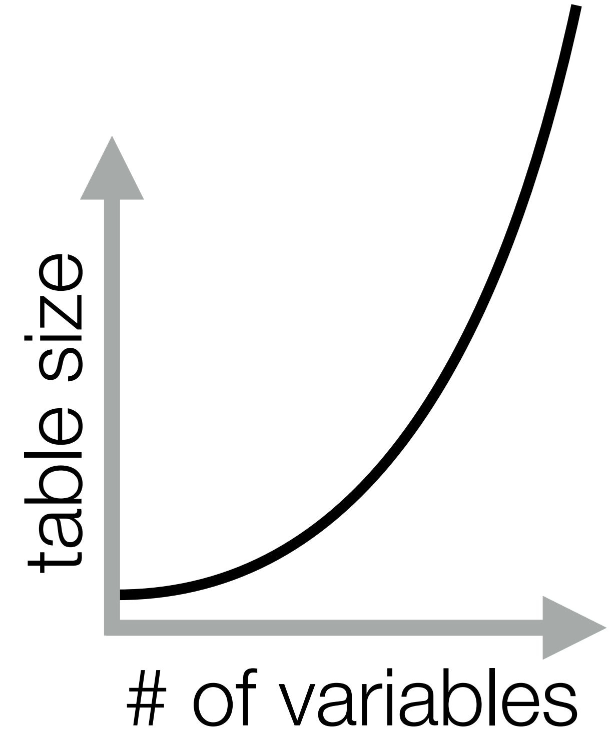
could we represent it using less than $2^n - 1$ numbers?



problems?

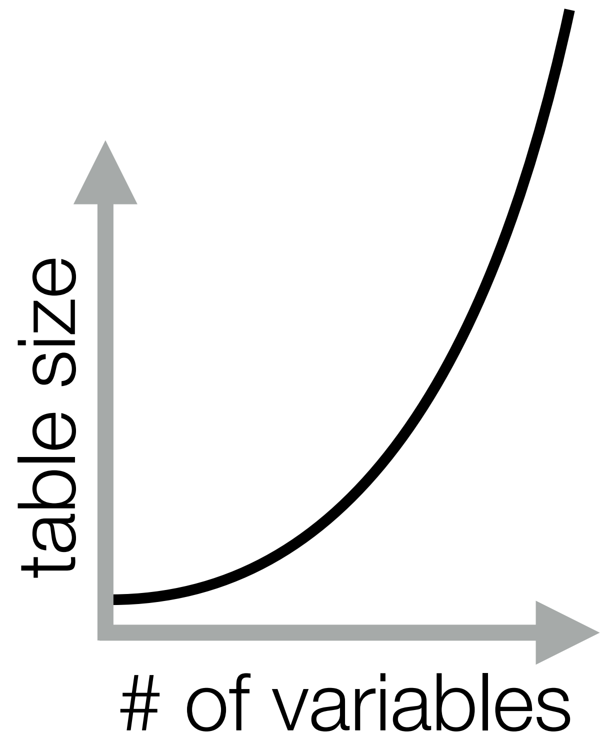
could we represent it using less than $2^n - 1$ numbers?

composition of CPTs?



problems?

could we represent it using less than $2^n - 1$ numbers?

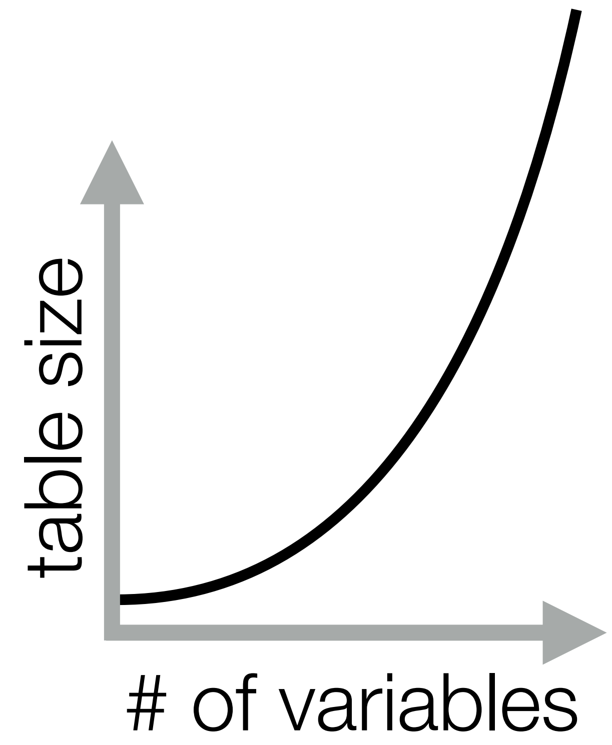


composition of CPTs?

$$P(x_1, x_2, x_3, ..., x_n) = P(x_1|x_2, x_3, ..., x_n)P(x_2|x_3, ..., x_n)...P(x_n)$$

problems?

could we represent it using less than $2^n - 1$ numbers?



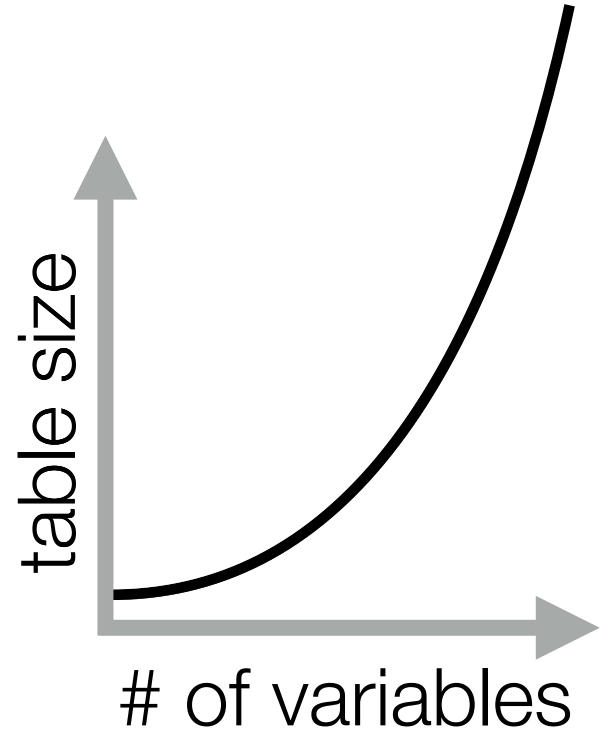
composition of CPTs?

$$P(x_1, x_2, x_3, ..., x_n) = P(x_1|x_2, x_3, ..., x_n)P(x_2|x_3, ..., x_n)...P(x_n)$$

$$P(x_1, x_2, x_3, ..., x_n) = P(x_1|x_2)P(x_2|x_3)...P(x_n)$$

problems?

could we represent it using less than $2^n - 1$ numbers?



composition of CPTs?

$$P(x_1, x_2, x_3, ..., x_n) = P(x_1|x_2, x_3, ..., x_n)P(x_2|x_3, ..., x_n)...P(x_n)$$

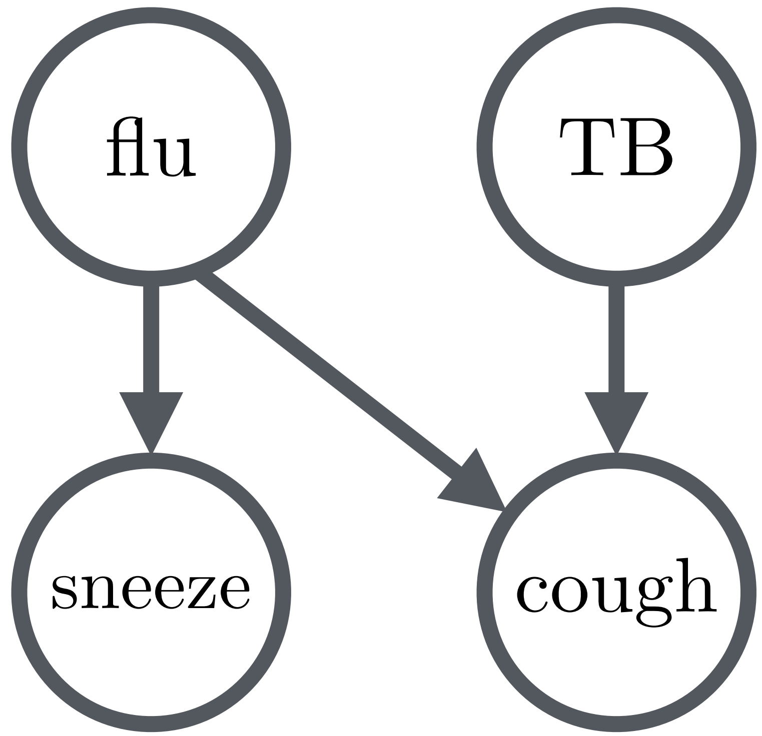
independencies

$$P(x_1, x_2, x_3, ..., x_n) = P(x_1|x_2)P(x_2|x_3)...P(x_n)$$

bayes net representation

flu=t
0.2

TB=t
0.1



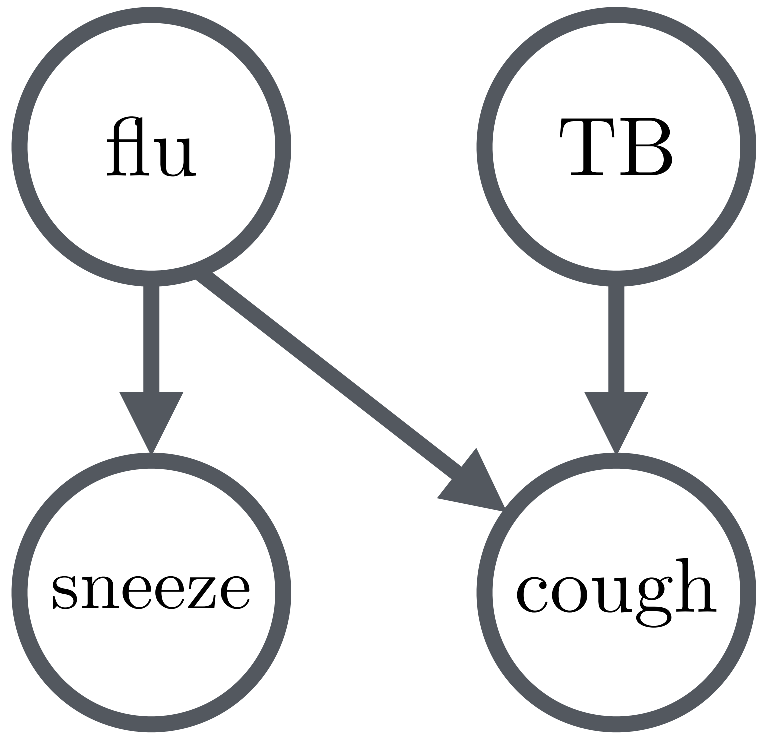
flu	sneeze=t
t	0.8
f	0.2

TB	flu	cough=t
t	t	0.9
t	f	0.8
f	t	0.75
f	f	0.1

bayes net representation

flu=t
0.2

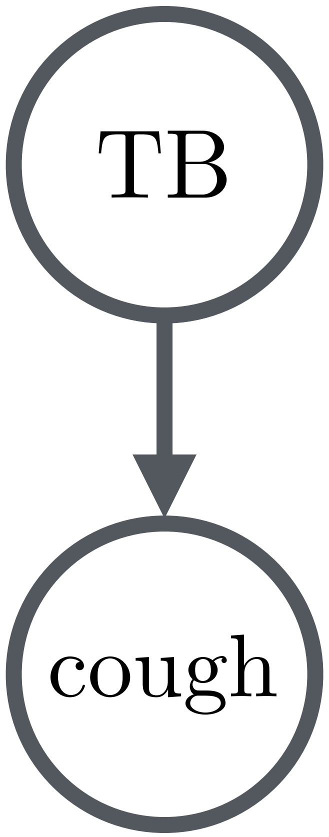
TB=t
0.1

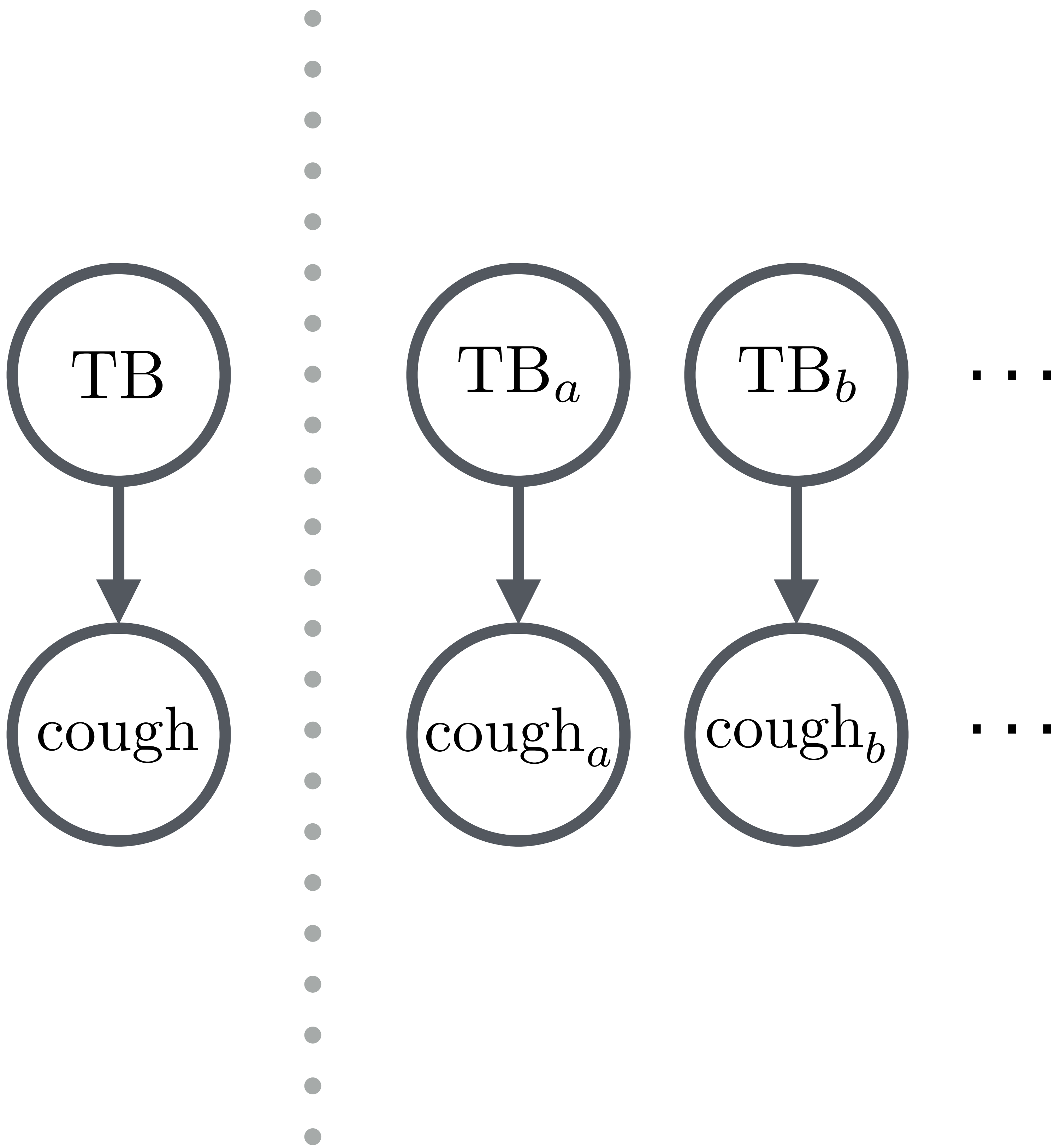


flu	sneeze=t
t	0.8
f	0.2

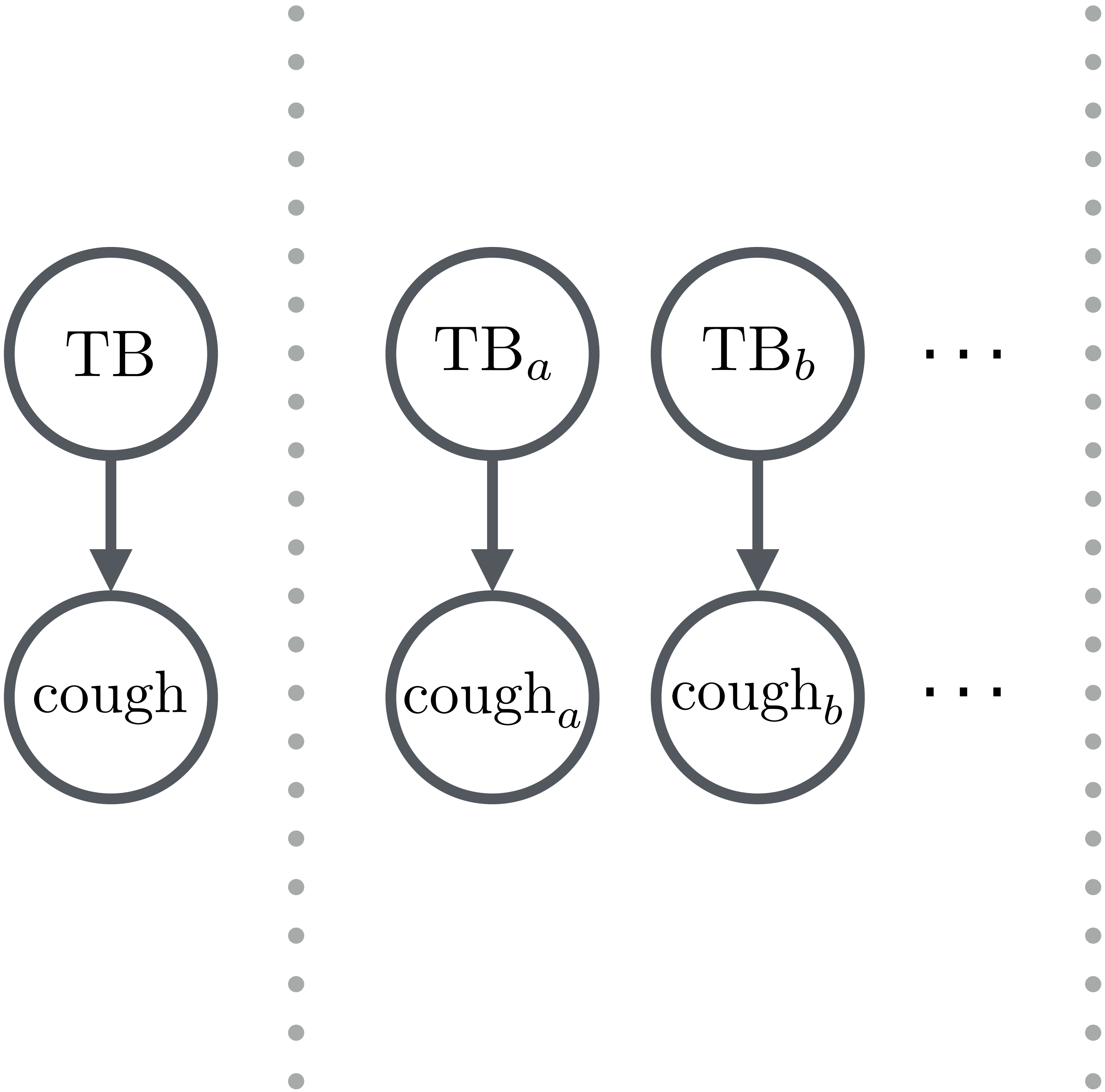
TB	flu	cough=t
t	t	0.9
t	f	0.8
f	t	0.75
f	f	0.1

is this enough?

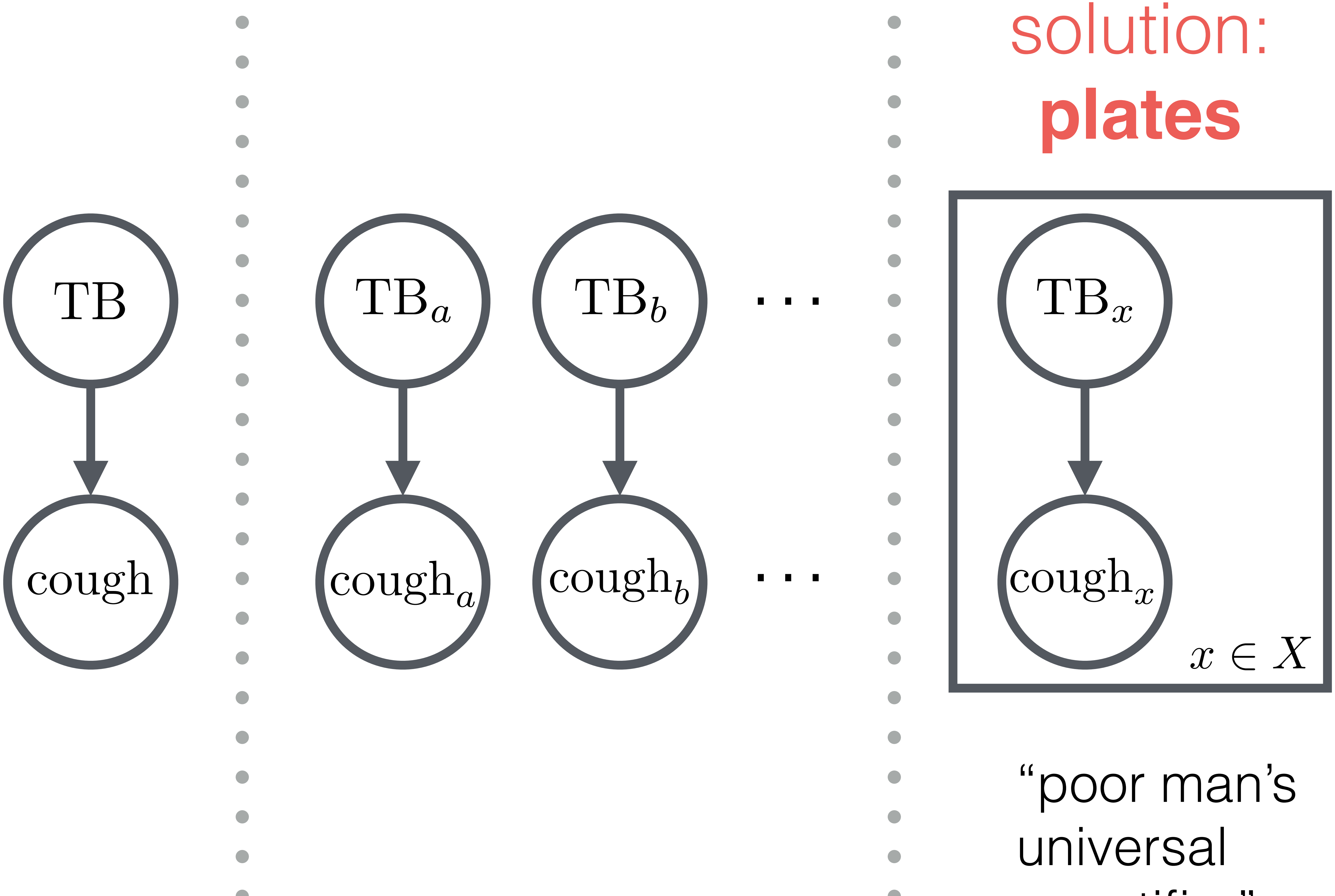

TB
cough


TB
cough
TB$_a$
cough$_a$
TB$_b$
cough$_b$
...
...

solution:
plates

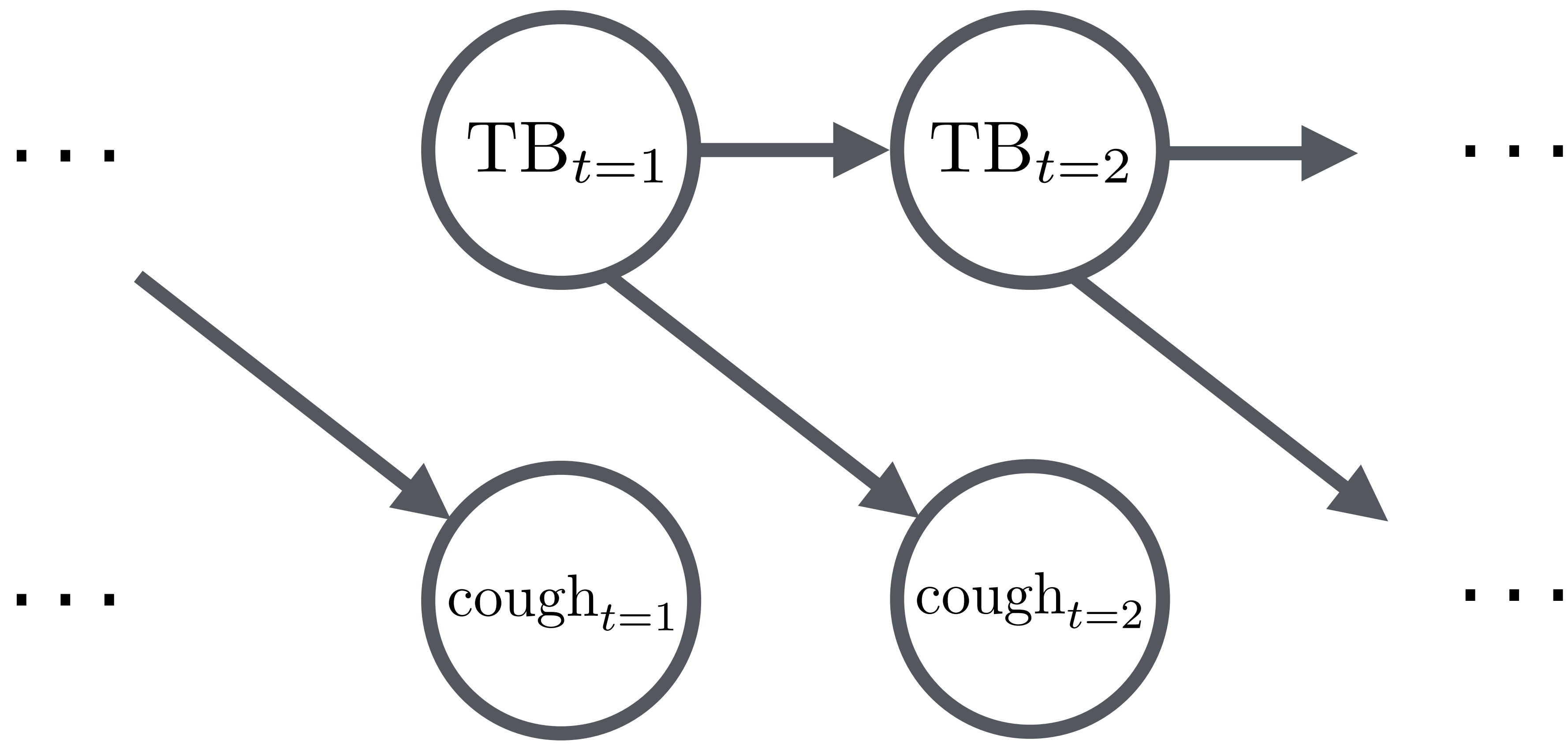


solution:
plates

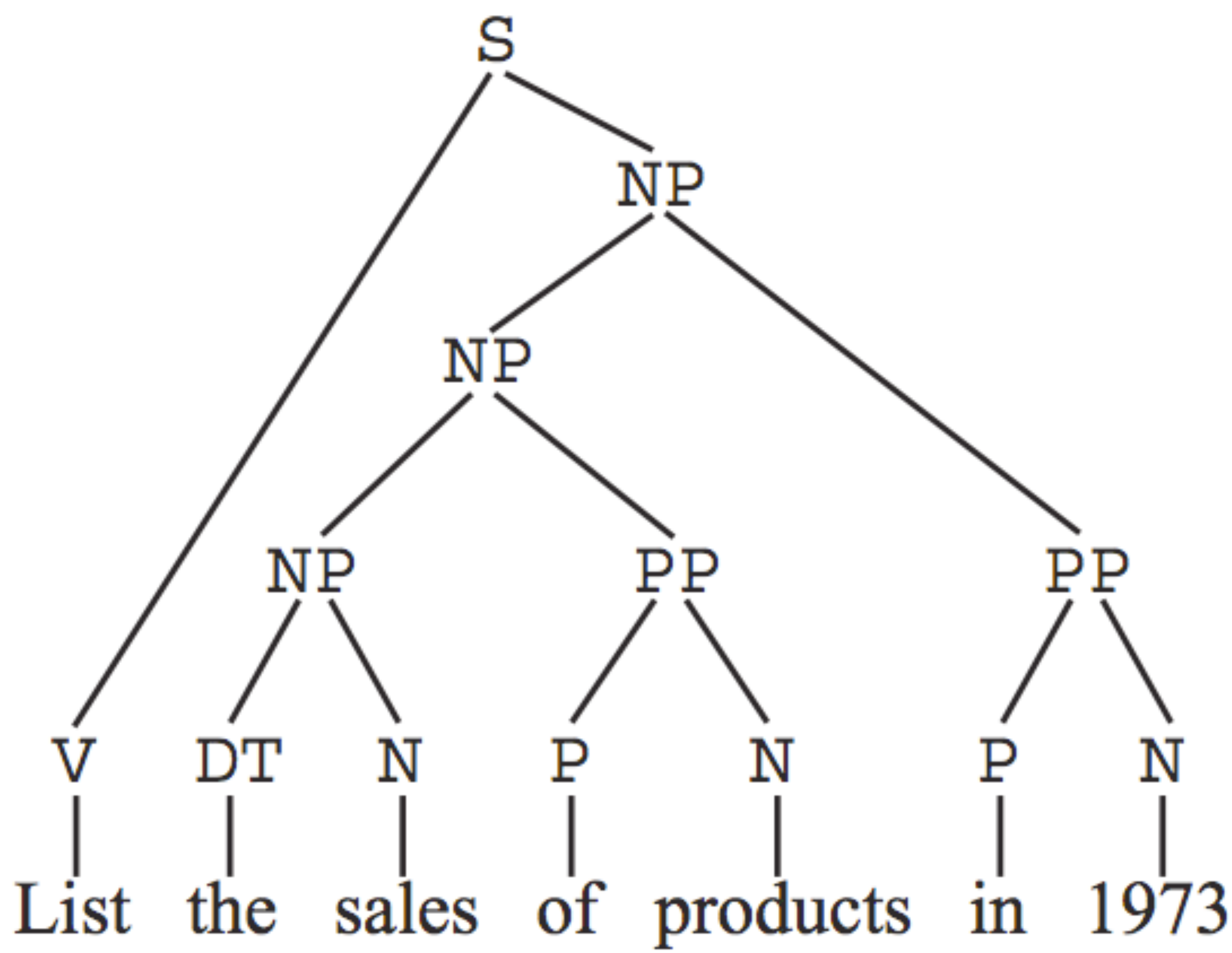


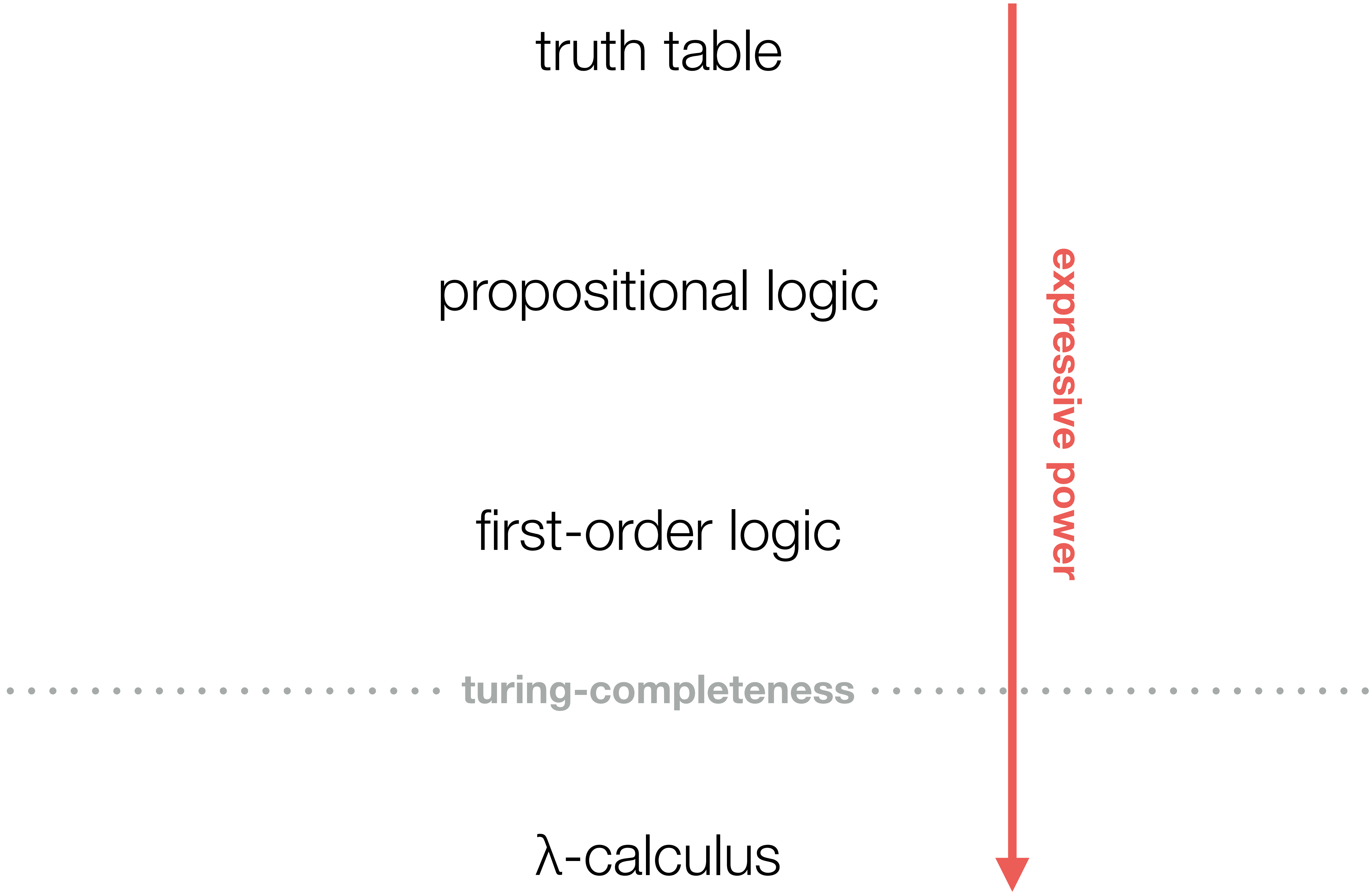
"poor man's universal quantifier"

TB now causes coughing *later*



PCFG




truth table
propositional logic
first-order logic
turing-completeness
λ-calculus
expressive power

truth table

propositional logic

first-order logic

turing-completeness

λ-calculus

truth table

joint probability table

propositional logic

first-order logic

turing-completeness

λ-calculus

truth table ↔ joint probability table

propositional logic ↔ bayes nets

first-order logic

· turing-completeness ·

λ-calculus

truth table	↔	joint probability table
propositional logic	↔	bayes nets
first-order logic		
turing-completeness		
λ-calculus	↔	higher-order probability?

truth table ↔ joint probability table

propositional logic ↔ bayes nets

first-order logic

· **turing-completeness** ·

λ-calculus ↔ ψλ -calculus

truth table	↔	joint probability table
propositional logic	↔	bayes nets
first-order logic		

turing-completeness

λ-calculus	↔	ψλ -calculus
UTM	↔	UPTM

λ-calculus

λ-calculus

```
(sin x)
```

λ-calculus

```
(sin x)
```

```
(+ x y)
```

λ-calculus

```
(sin x)
```

```
(+ x y)
```

```
(define double
  λ (x) (+ x x))
```

λ-calculus

```
(sin x)
```

```
(+ x y)
```

```
(define double
  λ (x) (+ x x))
```

```
(double 3)
```
→ 6

λ-calculus

```
(sin x)
```

```
(+ x y)
```

```
(define double
  λ (x) (+ x x))
```

```
(double 3)
```
→ 6

```
(define repeat
  λ (f) (λ (x) (f (f x))))
```

λ-calculus

```
(sin x)
```

```
(+ x y)
```

```
(define double
  λ (x) (+ x x))
```

```
(double 3)
```

→ 6

```
(define repeat
  λ (f) (λ (x) (f (f x))))
```

```
((repeat double) 3)
```

↓

12

λ-calculus

```
(sin x)
```

```
(+ x y)
```

```
(define double
  λ (x) (+ x x))
```

```
(double 3)
```

→ 6

```
(define repeat
  λ (f) (λ (x) (f (f x))))
```

```
((repeat double) 3)
```

↓

12

```
(define 2nd-derivative
   (repeat derivative))
```

ψλ-calculus

ψλ-calculus

+ a random choice operator

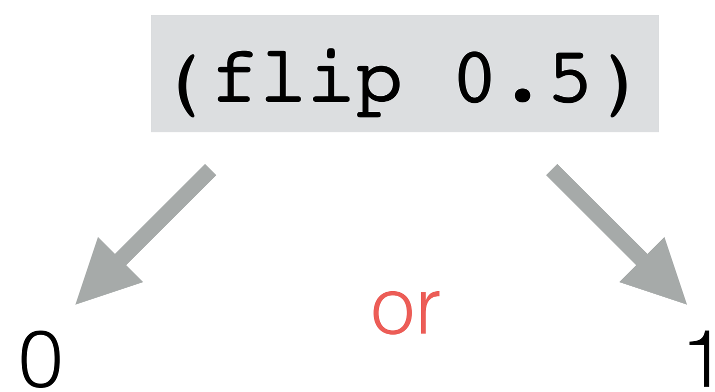
ψλ-calculus

+ a random choice operator

```
(flip 0.5)
```

ψλ-calculus

+ a random choice operator



```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

→ 1

```
(define a (flip 0.3))   → 1
(define b (flip 0.3))   → 0
(define c (flip 0.3))
(+ a b c)
```

```
(define a (flip 0.3))  → 1
(define b (flip 0.3))  → 0
(define c (flip 0.3))  → 1
(+ a b c)
```

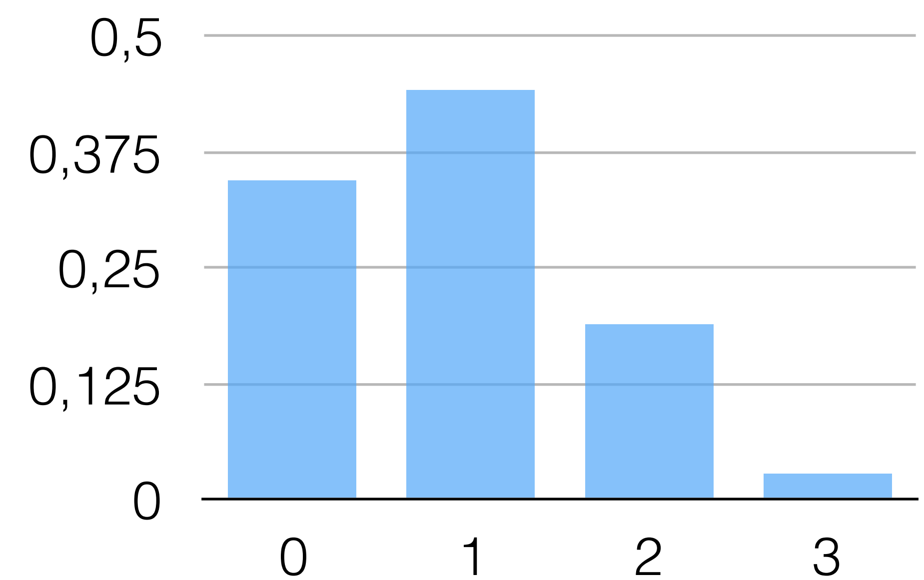
`(define a (flip 0.3))`	1
`(define b (flip 0.3))`	0
`(define c (flip 0.3))`	1
`(+ a b c)`	2

`(define a (flip 0.3))`	1 0
`(define b (flip 0.3))`	0 0
`(define c (flip 0.3))`	1 0
`(+ a b c)`	2 0

`(define a (flip 0.3))` →	1 0 0
`(define b (flip 0.3))` →	0 0 0
`(define c (flip 0.3))` →	1 0 1
`(+ a b c)` →	2 0 1

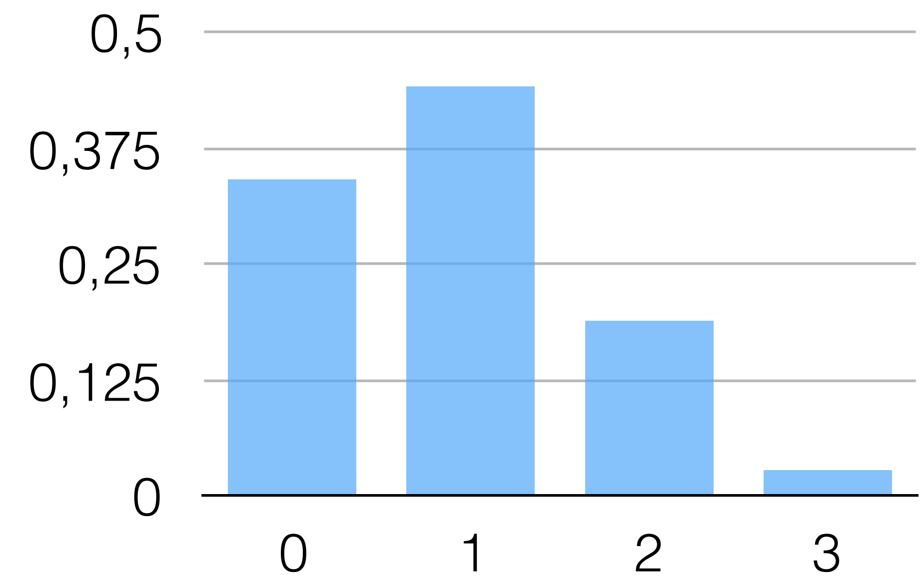
```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

1 0 0
0 0 0
1 0 1
2 0 1



```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

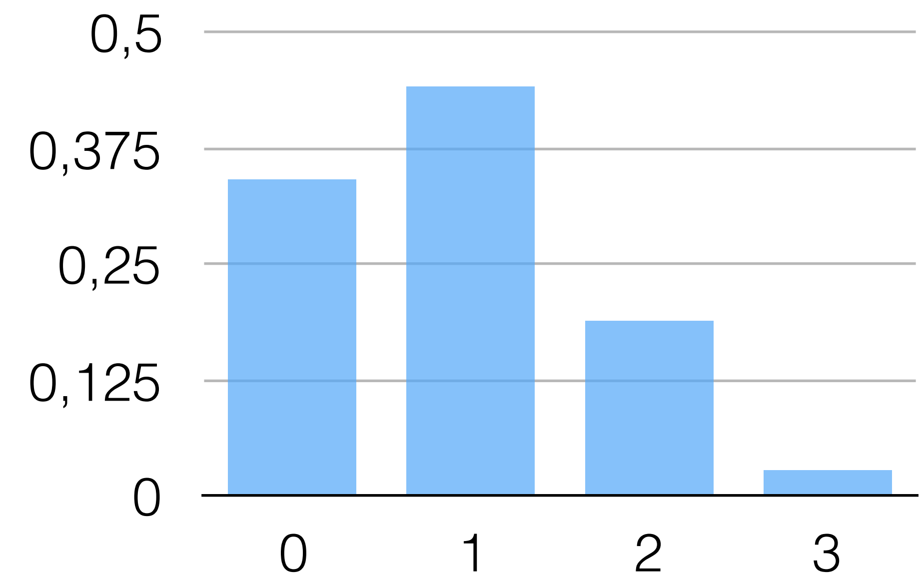
1	0	0	1
0	0	0	0
1	0	1	2
2	0	1	3



$$P(n) = \binom{3}{n} 0.3^n 0.7^{3-n}$$

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

1 0 0
0 0 0
1 0 1
2 0 1



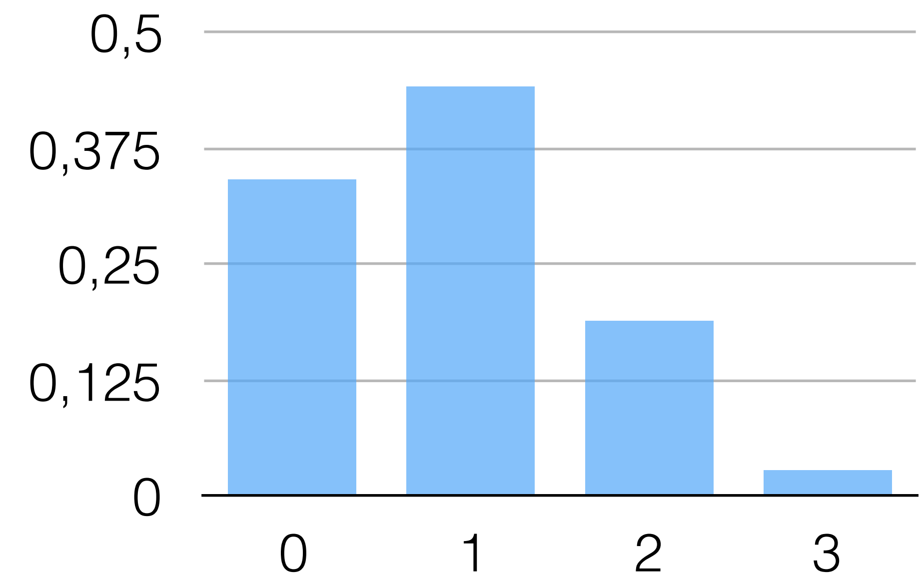
$$P(n) = \binom{3}{n} 0.3^n 0.7^{3-n}$$

distribution semantics

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

1 0 0
0 0 0
1 0 1
2 0 1

sampling semantics

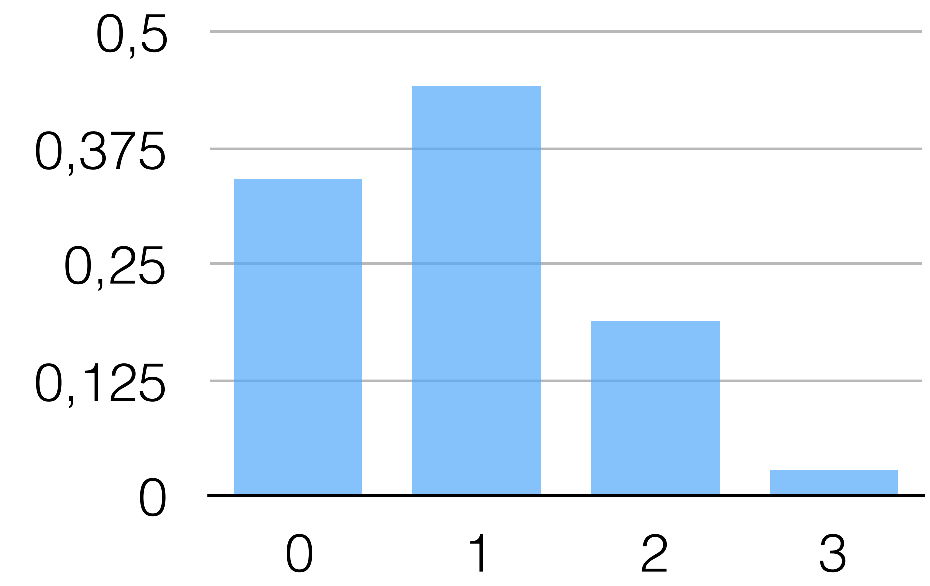


$$P(n) = \binom{3}{n} 0.3^n 0.7^{3-n}$$

distribution semantics

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

1 0 0
0 0 0
1 0 1
2 0 1



$$P(n) = \binom{3}{n} 0.3^n 0.7^{3-n}$$

sampling semantics ⟷ **distribution semantics**

theorem

any computable distribution can be represented by a Church expression

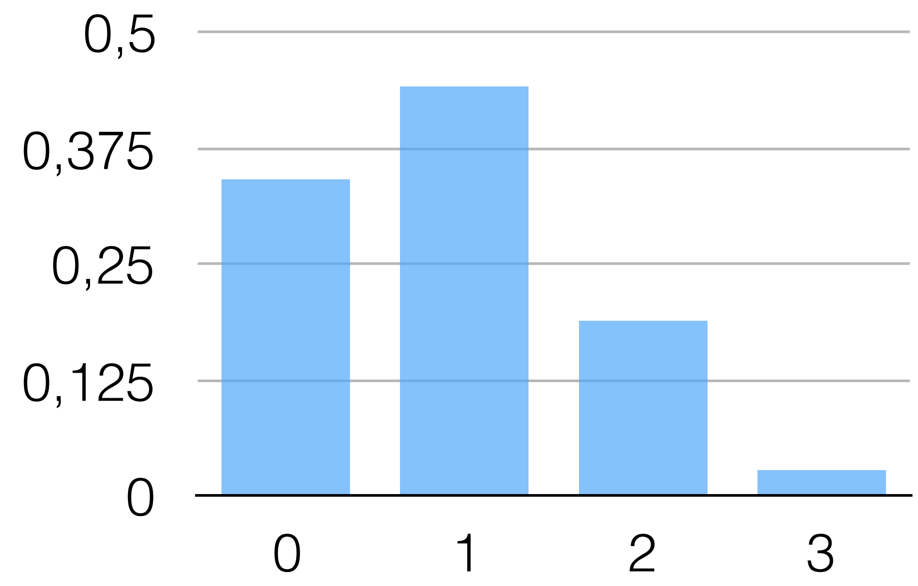
```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

→ 1 0 0

→ 0 0 0

→ 1 0 1

→ 2 0 1



$$P(n) = \binom{3}{n} 0.3^n 0.7^{3-n}$$

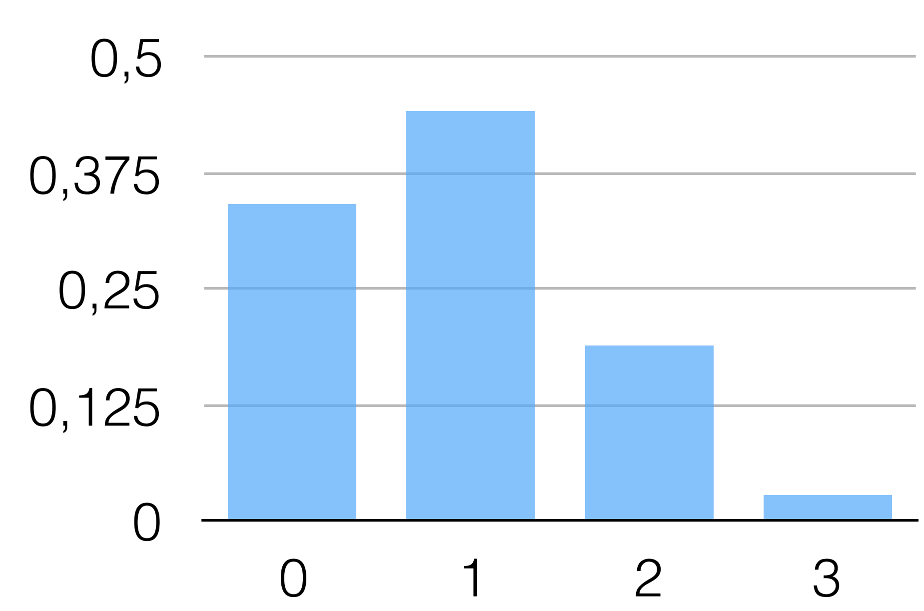
sampling semantics ↔ **distribution semantics**

theorem

any computable distribution can be represented by a Church expression

& any (a.s.) halting Church expression can be represented by a computable distribution

```
(define a (flip 0.3))   → 1 0 0
(define b (flip 0.3))   → 0 0 0
(define c (flip 0.3))   → 1 0 1
(+ a b c)               → 2 0 1
```



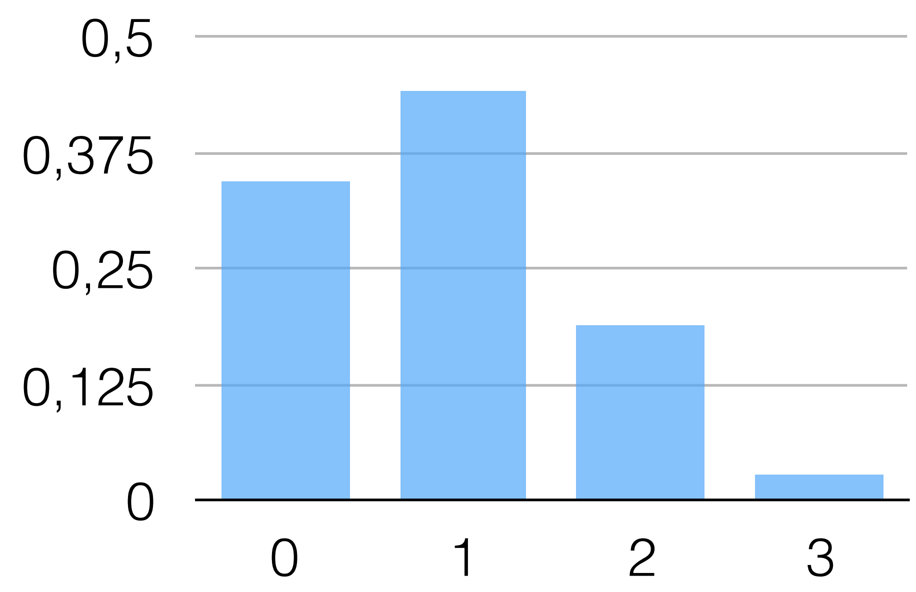
$$P(n) = \binom{3}{n} 0.3^n 0.7^{3-n}$$

sampling semantics ⟷ **distribution semantics**

LISP / λ-calculus = CHURCH / ψλ-calculus

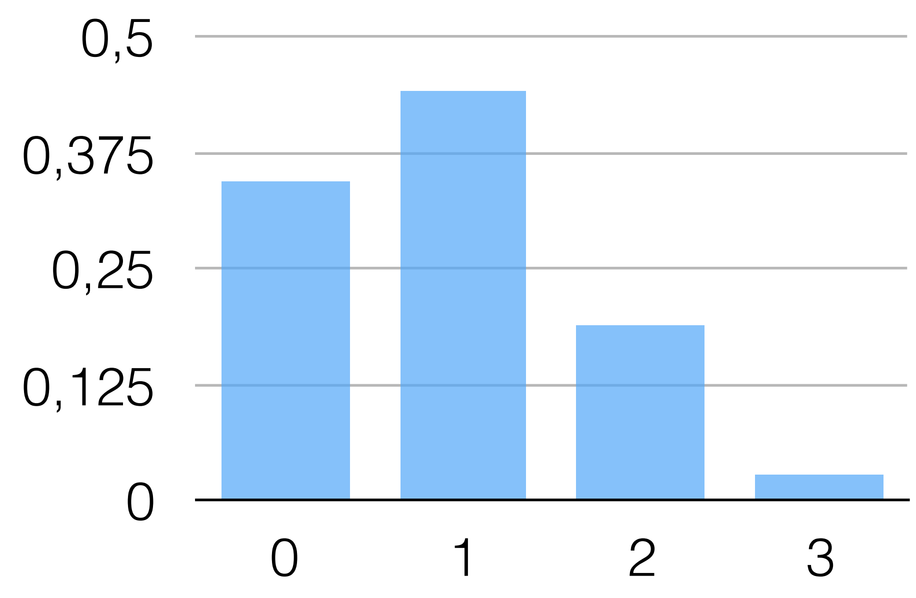
inference in CHURCH

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```



inference in CHURCH

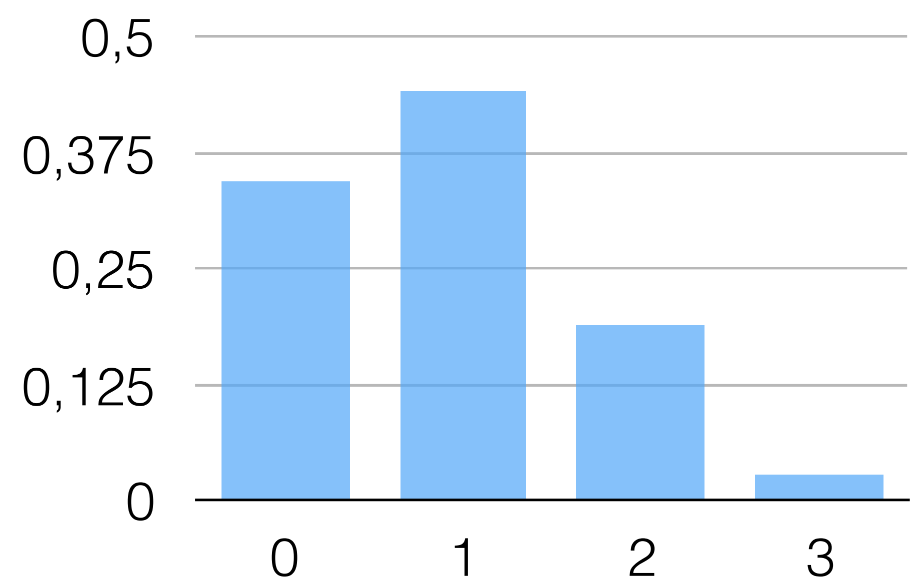
```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```




```
(query
  (define a (flip 0.3))
  (define b (flip 0.3))
  (define c (flip 0.3))
  (+ a b c)
  (= (+ a b) 1))
```

inference in CHURCH

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

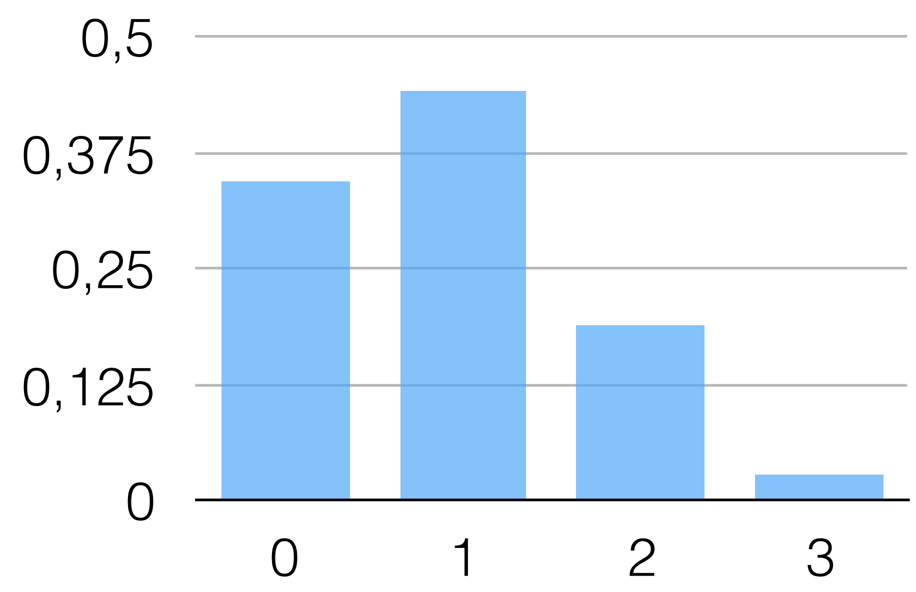



```
(query
  (define a (flip 0.3))
  (define b (flip 0.3))
  (define c (flip 0.3))
  (+ a b c)
  (= (+ a b) 1))
```

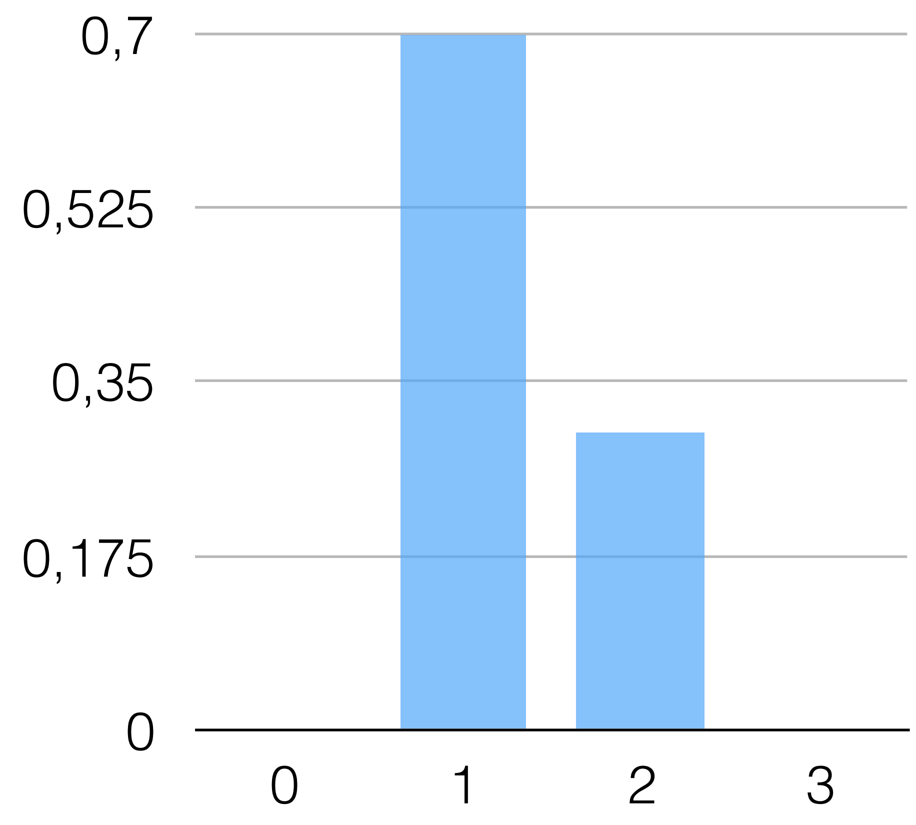
must be true

inference in CHURCH

```
  (define a (flip 0.3))
  (define b (flip 0.3))
  (define c (flip 0.3))
  (+ a b c)
```

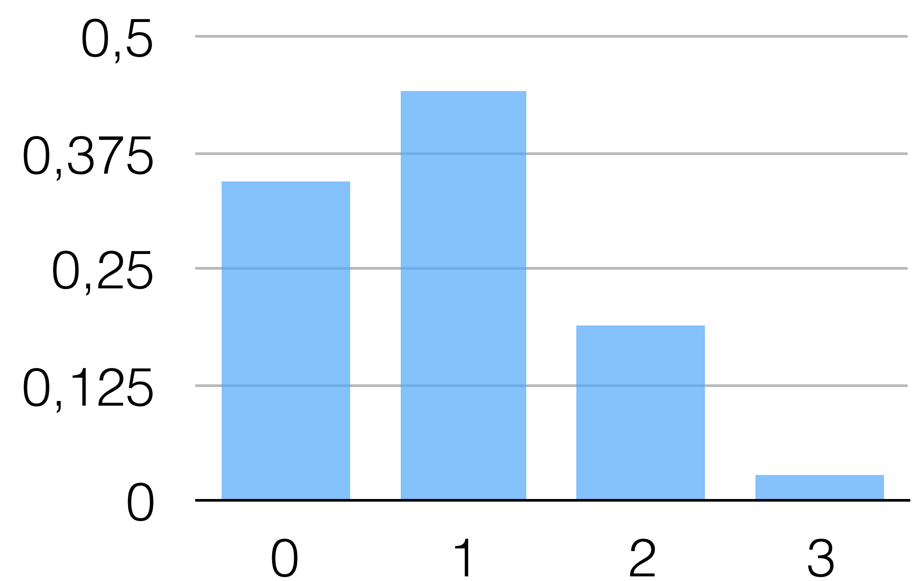



```
(query
  (define a (flip 0.3))
  (define b (flip 0.3))
  (define c (flip 0.3))
  (+ a b c)
  (= (+ a b) 1))
```
must be true



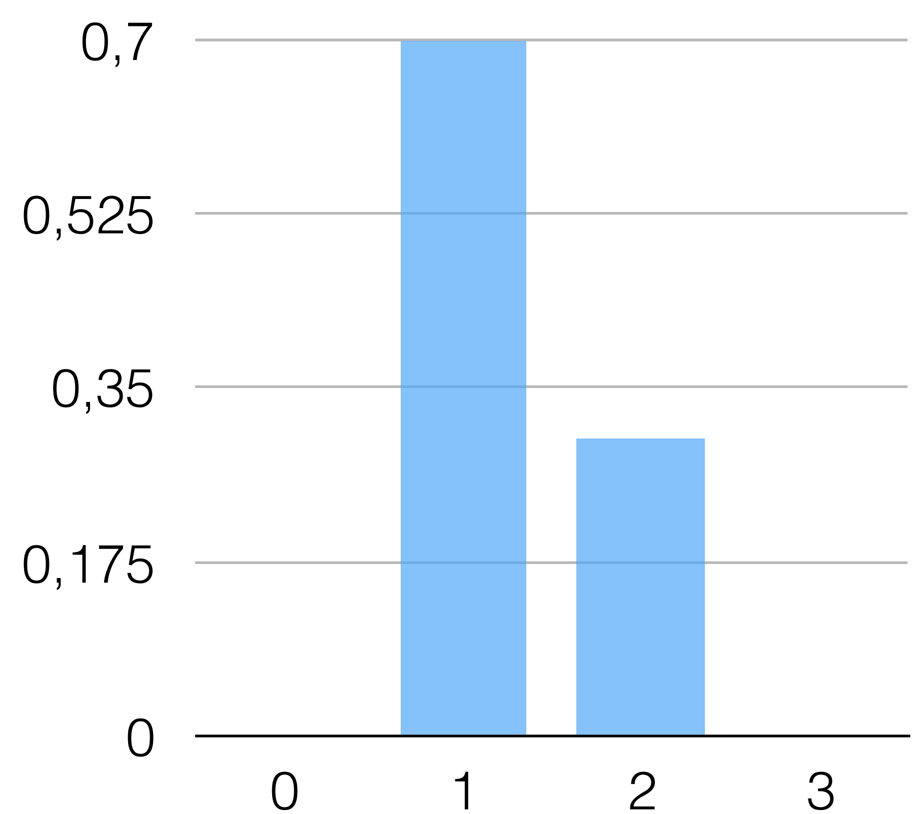
inference in CHURCH

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```




```
(query
  (define a (flip 0.3))
  (define b (flip 0.3))
  (define c (flip 0.3))
  (+ a b c)
  (= (+ a b) 1))
```

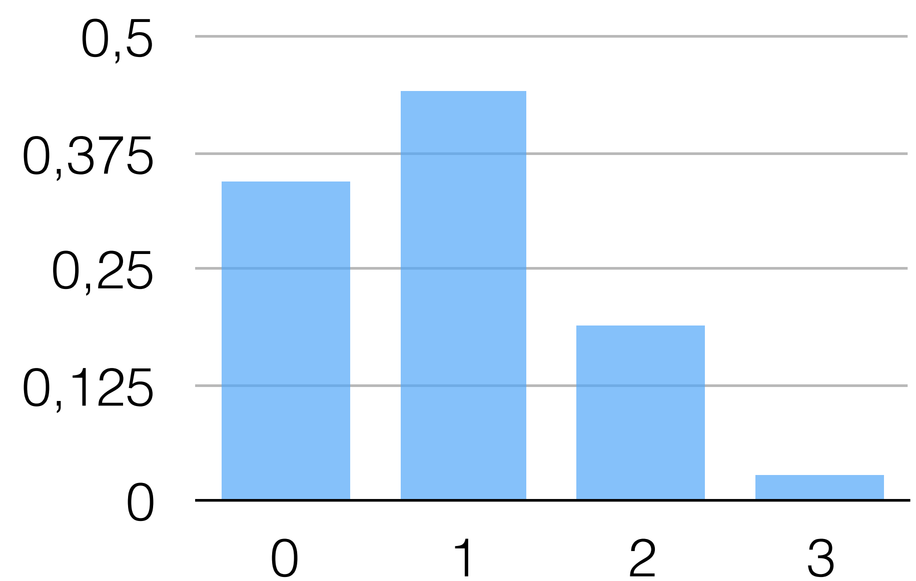
must be true



$$P(a + b + c | a + b = 1)$$

inference in CHURCH

```
(define a (flip 0.3))
(define b (flip 0.3))
(define c (flip 0.3))
(+ a b c)
```

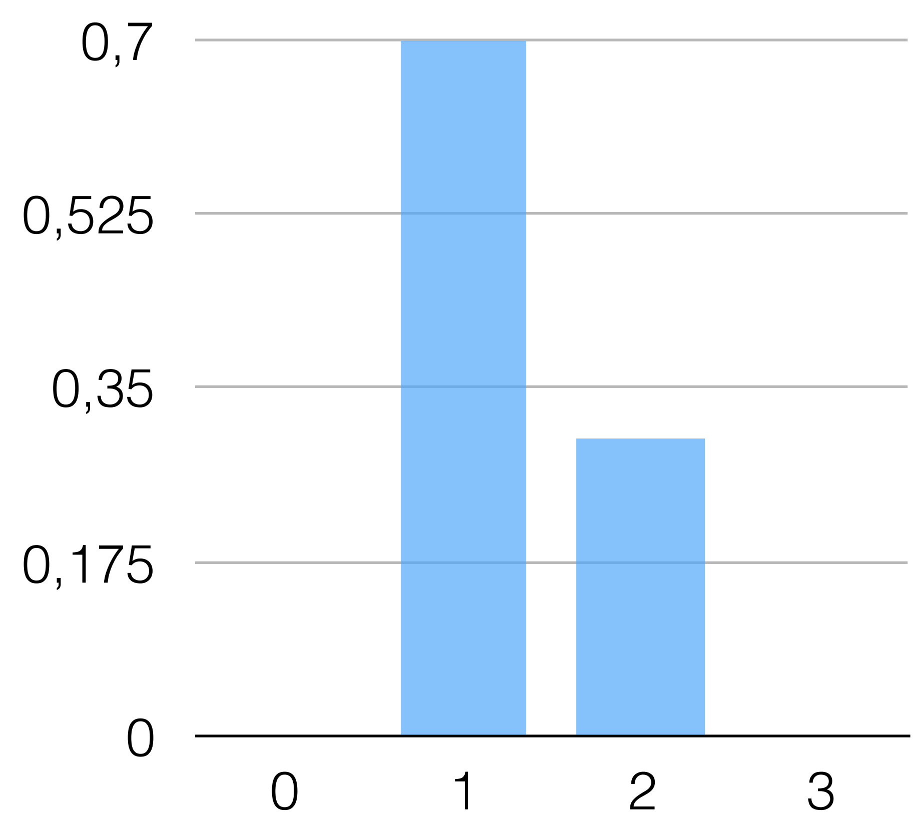


rejection sampling

```
(query
  (define a (flip 0.3))
  (define b (flip 0.3))
  (define c (flip 0.3))
  (+ a b c)
  (= (+ a b) 1))
```

must be true

$$P(a+b+c|a+b=1)$$



Probabilistic language of thought hypothesis (formal version): Concepts are stochastic functions. Hence they represent uncertainty, compose naturally, and support probabilistic inference.

- implications for learning, natural language, theory of mind, intuitive theories and mental simulations, etc.

learning

- concept learning = program induction = density estimation = inference

learning

- concept learning = program induction = density estimation = inference

- concepts are like software libraries for a programming language:
the hypothesis space stays the same but the formation of new concepts makes certain hypotheses simpler (higher prior)

learning

- concept learning = program induction = density estimation = inference

- concepts are like software libraries for a programming language:
the hypothesis space stays the same but the formation of new concepts makes certain hypotheses simpler (higher prior)

- information theoretic view: construct a language where events in the world are shorter to describe