Extending ProbLog with Continuous Distributions

Bernd Gutmann, Manfred Jaeger, Luc De Raedt

ILP 2010 - 30th June 2010

KATHOLIEKE UNIVERSITEIT LEUVEN

AALBORG UNIVERSITET

DTAI
MACHINE LEARNING

Hybrid ProbLog

- Combine
  - Logic programming
  - Probabilities
  - Continuous distributions
- Allow for exact inference
  P(query q has a proof)

Continuous Distributions

- occur in real-world domains
- when used, exact inference becomes hard
- sampling is not always desirable
- Question
  - What restrictions are necessary such that exact inference becomes tractable/possible?

Example

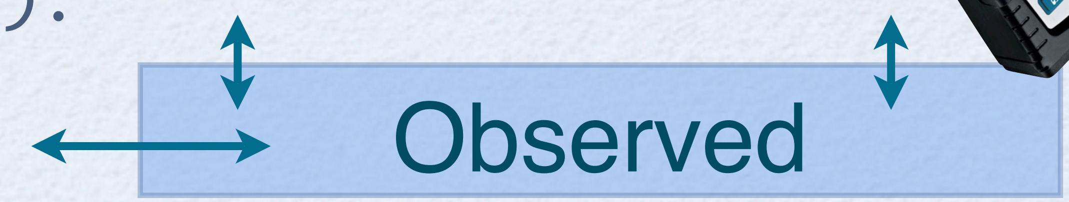
Example

Example

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```

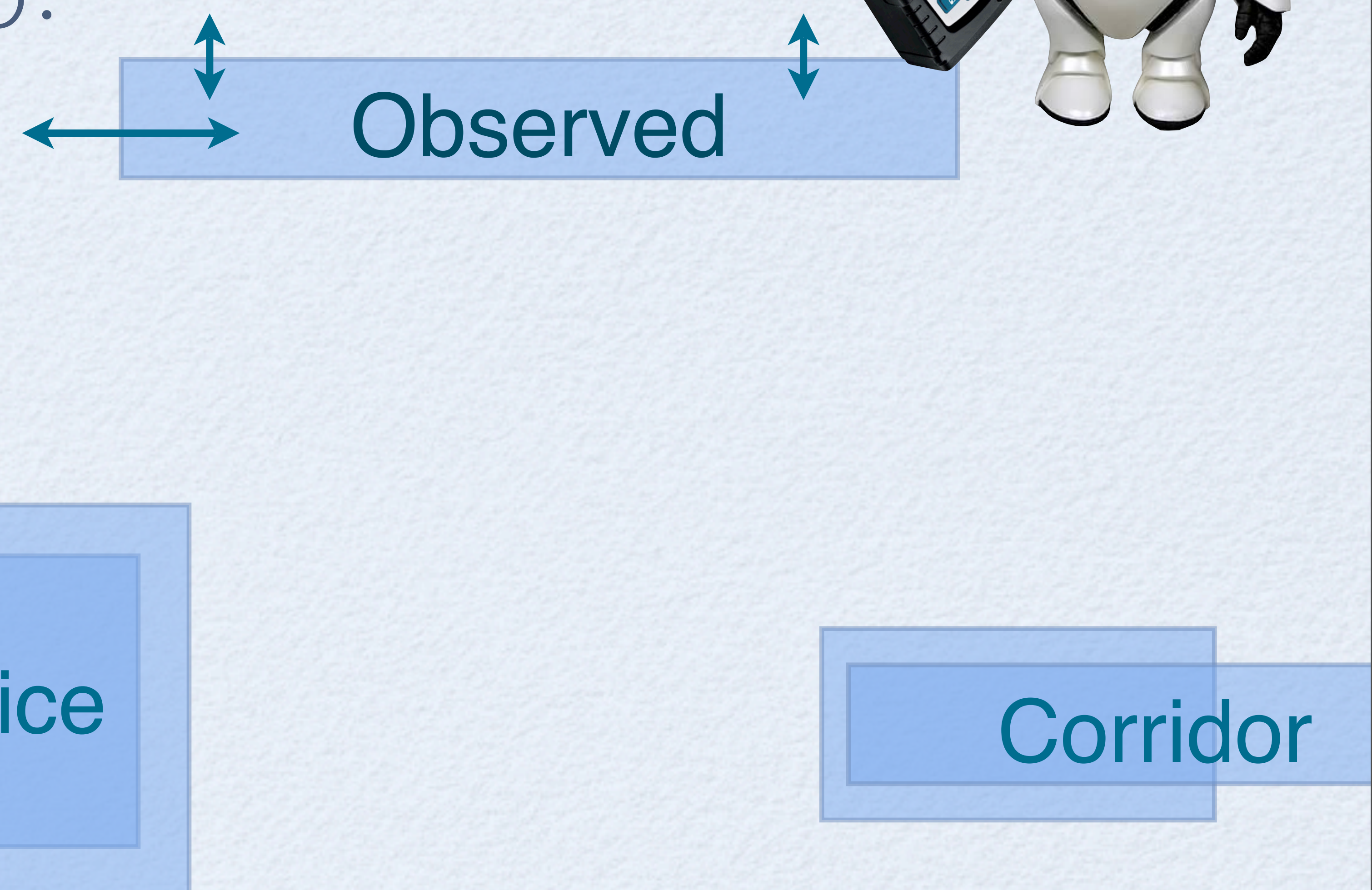
Example

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```



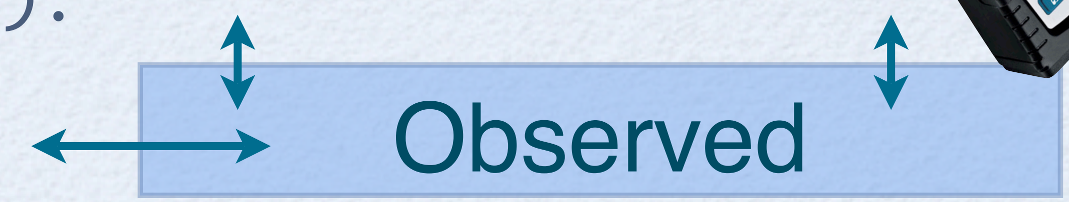
Example

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```



Example

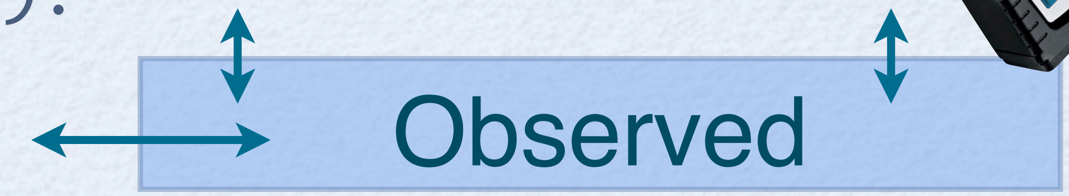
```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```




```
in_office :-
   width(W), length(L),
   in_interval(W,2,4),
   in_interval(L,2,4).
```

Example

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```




```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```


```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

Example


```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

Observed

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

Office

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

Corridor

Example


```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).

0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

Observed

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

Office

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

Corridor

```
room_has_window :-
   in_office,
   office_has_window.
```

Example

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

Observed

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

Office

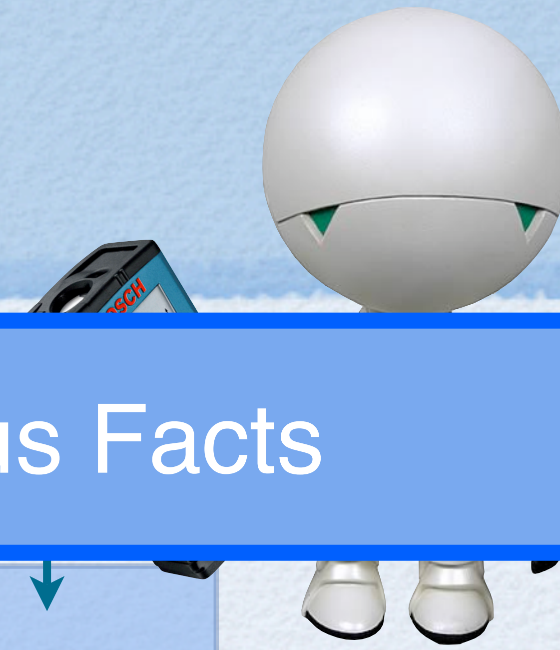
```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

Corridor

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Example



```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```

Continuous Facts

```
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

Observed

```
in_office :-
    width(W), length(L),
    in_interval(W,2,4),
    in_interval(L,2,4).
```

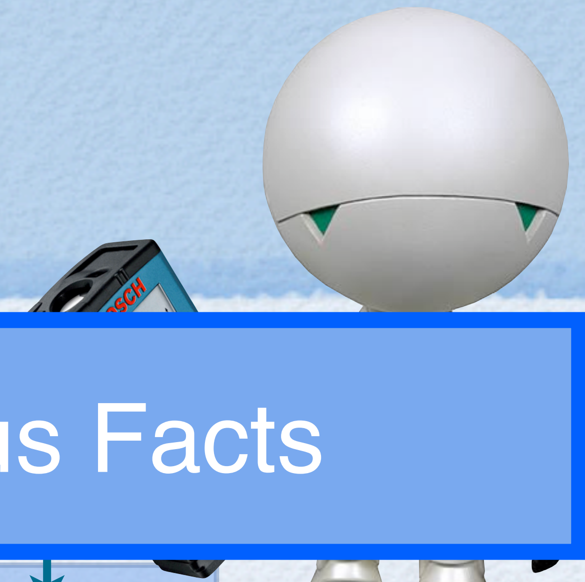
Office

```
in_corridor :-
    width(W), length(L),
    below(W,2.5),
    above(L,3).
```

Corridor

```
room_has_window :-
    in_office,
    office_has_window.

room_has_window :-
    in_corridor,
    corridor_has_window.
```

Example



```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```

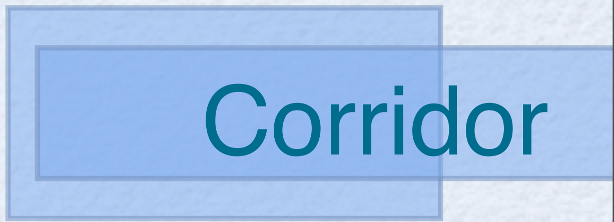
Continuous Facts

```
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

Probabilistic Facts

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

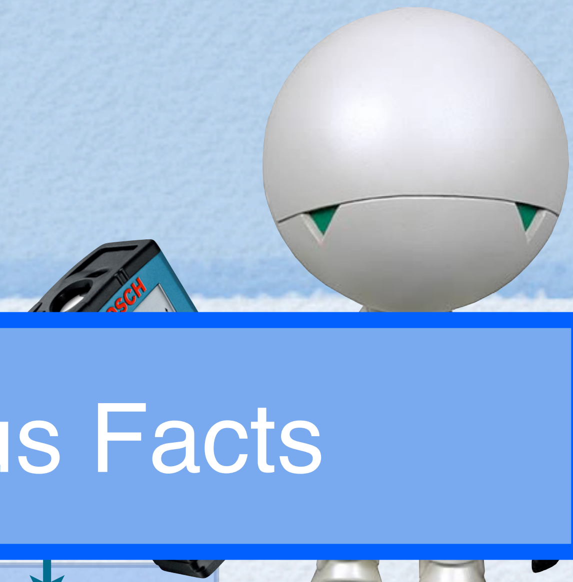

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```




```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Example



Continuous Facts

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
```

Probabilistic Facts

```
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

Background Knowledge

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

room_has_window :-
  in_office,
  office_has_window.
```

Office

```
  above(L,3).

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Corridor

Example

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

room_has_window :-
  in_office,
  office_has_window.
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Example

P(in_corridor) = 0.67

P(in_office) = 0.01

P(in_corridor | room_has_window) = 0.30

P(in_office | room_has_window) = 0.96

P(length(L),below(L,5))=0.07

P(length(L),below(L,5) | room_has_window)=0.80

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.

in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Outline

- Introduction
- **Hybrid Problog**
- Inference
- Learning
- Increase Expressivity

Syntax of Hybrid ProbLog

- probabilistic facts

```
0.8 :: office_has_window
0.8 :: office_has_window(Room)
```

- continuous facts

```
(L, gaussian(9,3)) :: length(L)
(L, gaussian(9,3)) :: length(L,Room)
(X, uniform(0,10)) :: position(X)
```

- comparison operators

```
... length(L), above(L,3)...
... length(L), below(L,3)...
... length(L), in_interval(L,2,4)...
```

Syntax of Hybrid ProbLog

- negation on ground probabilistic facts

  `problog_not(office_has_window)`

- negation on comparison operators

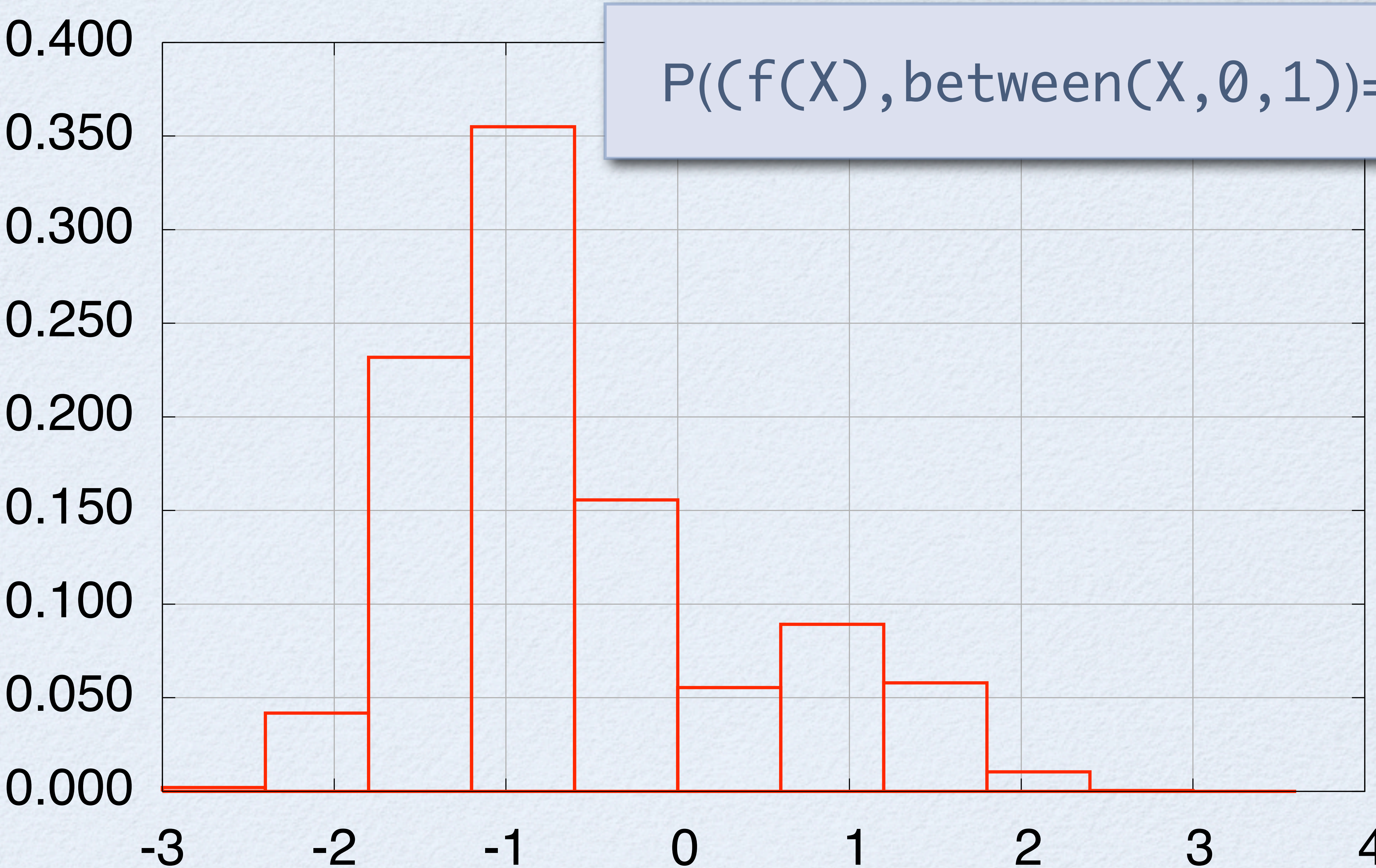
  `length(L), problog_not(in_interval(L,2,4)`

- Not allowed
  - functions
    `length(L), L2 is L*10`
  - comparing two values
    `length(L), width(W), below(W,L)`
  - multivariate distributions

Mixture of Gaussians

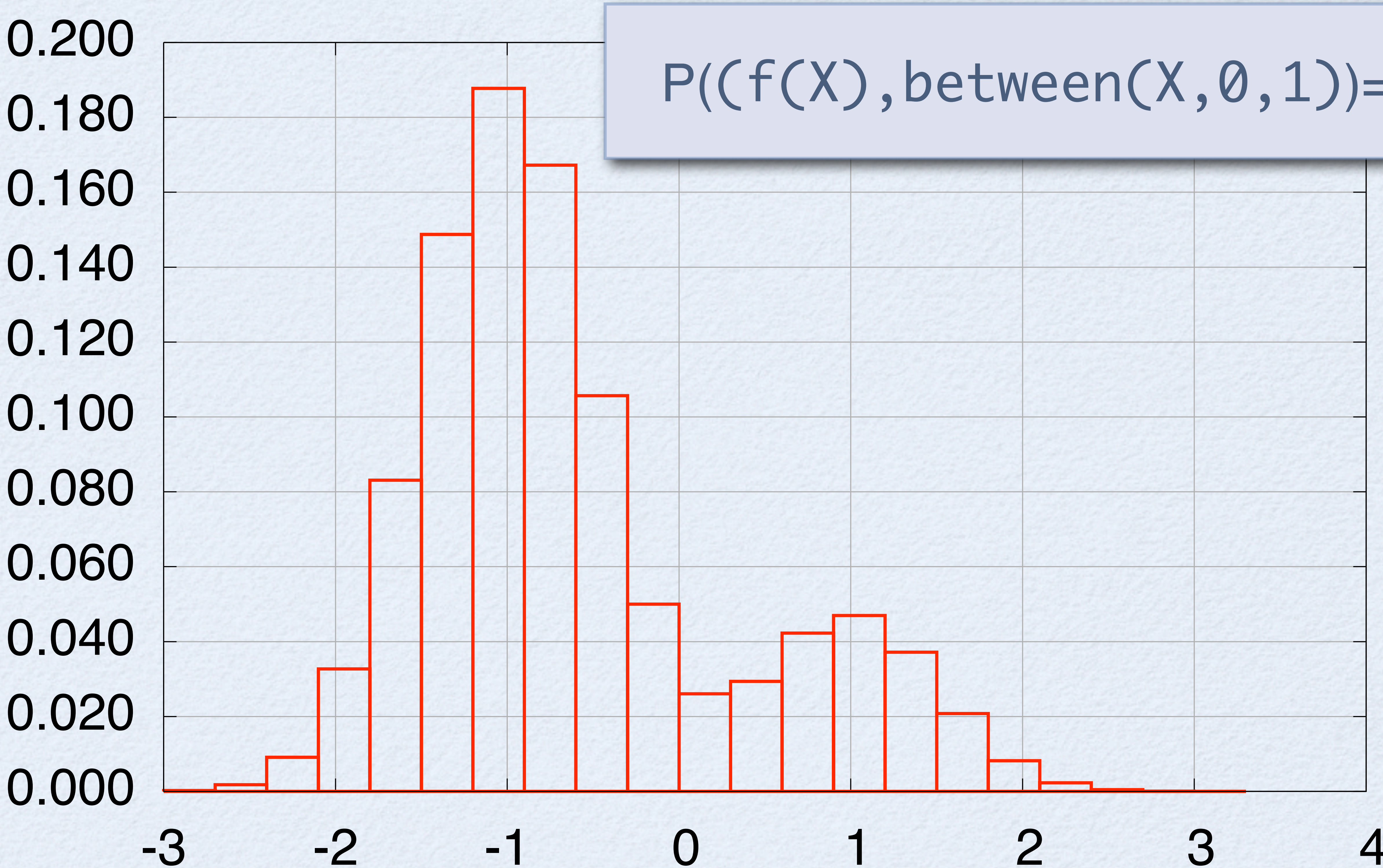
```
0.8 :: flip.

(X, gaussian(-1,1)) :: f1(X).

(X, gaussian(1,1)) :: f2(X).


f(X) :- flip, f1(X).

f(X) :- problog_not(flip), f2(X).
```

10 Intervals



P((f(X),between(X,0,1))=?

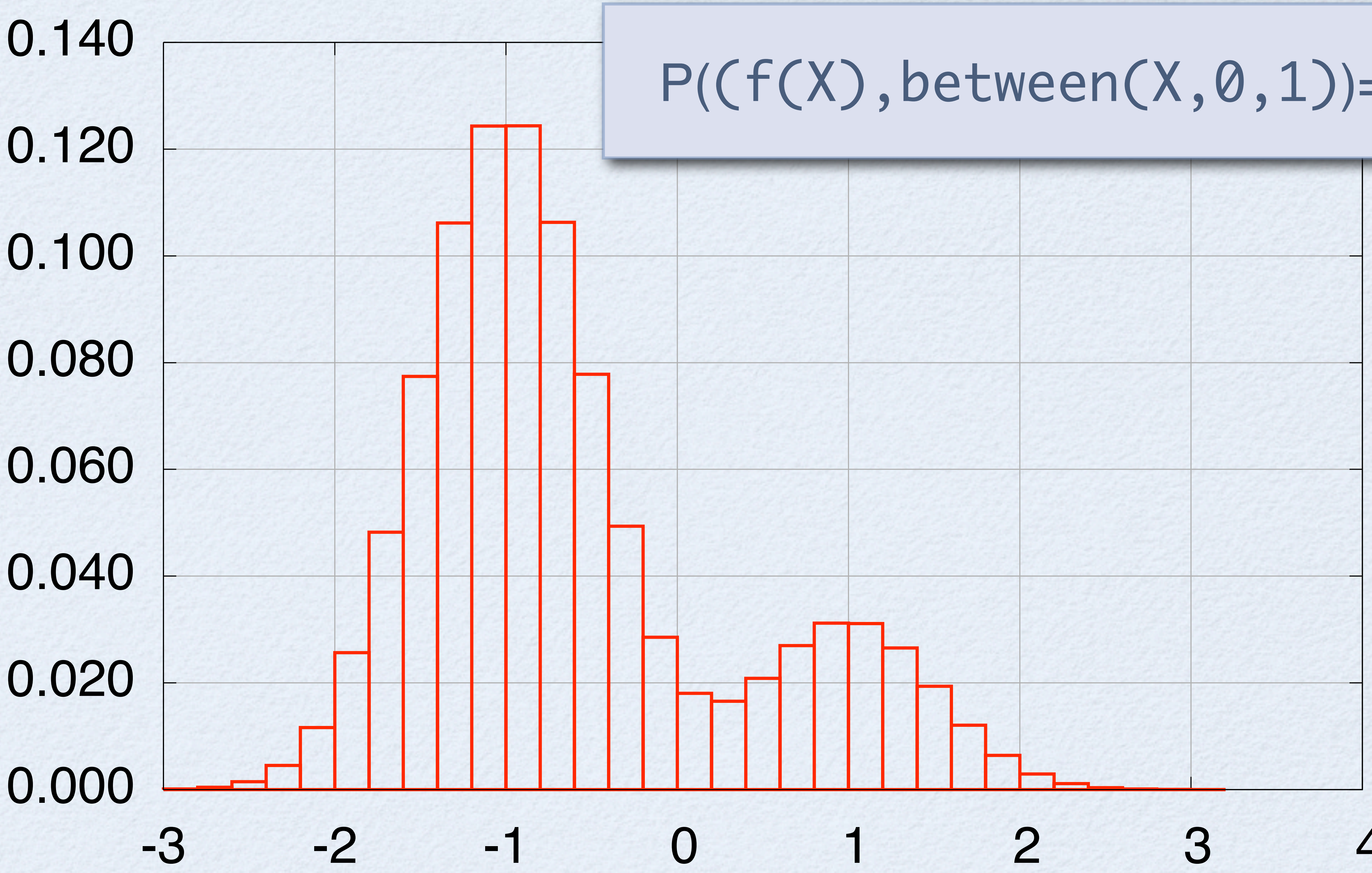
20 Intervals



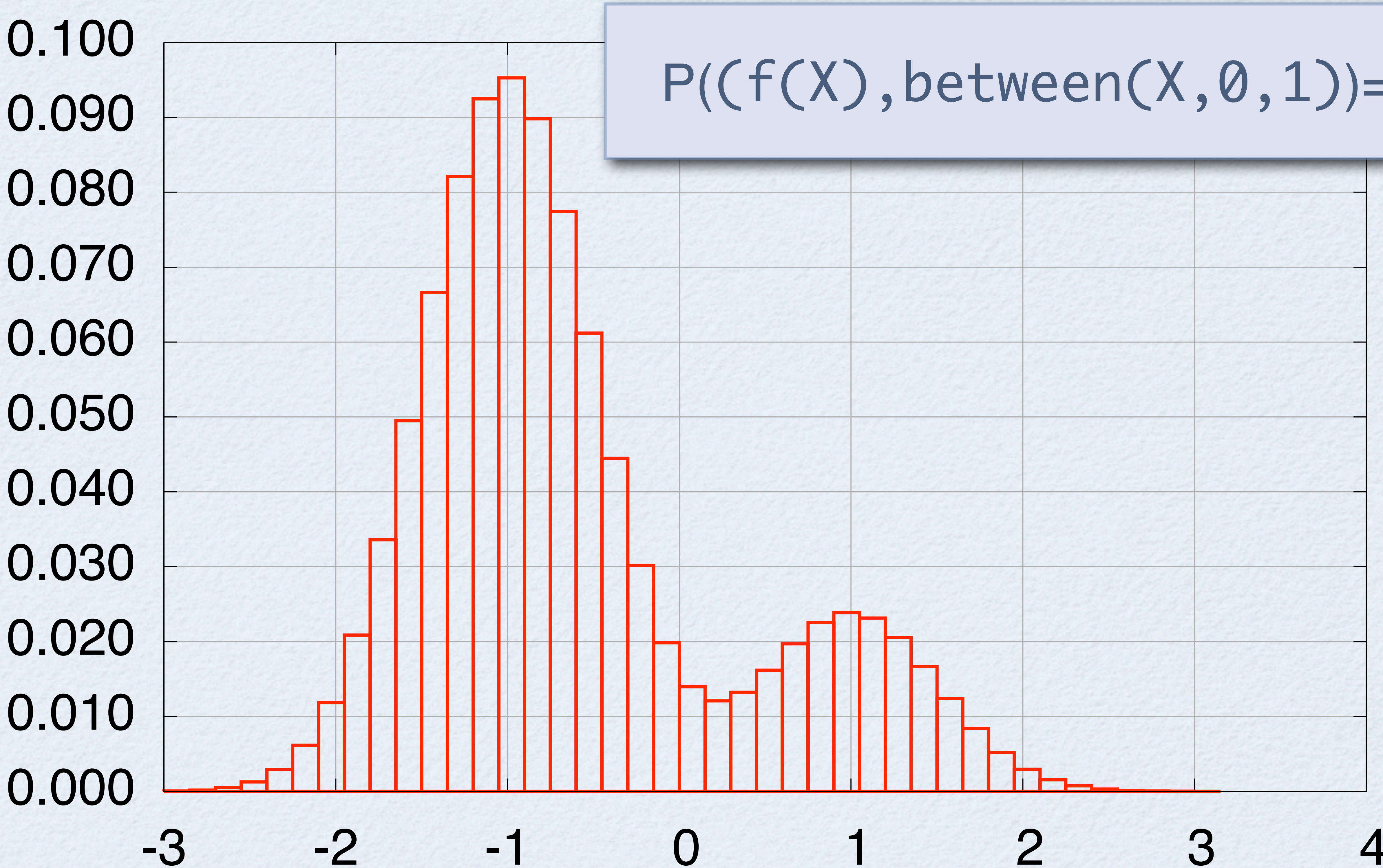
P((f(X),between(X,0,1))=?

30 Intervals

P((f(X),between(X,0,1))=?

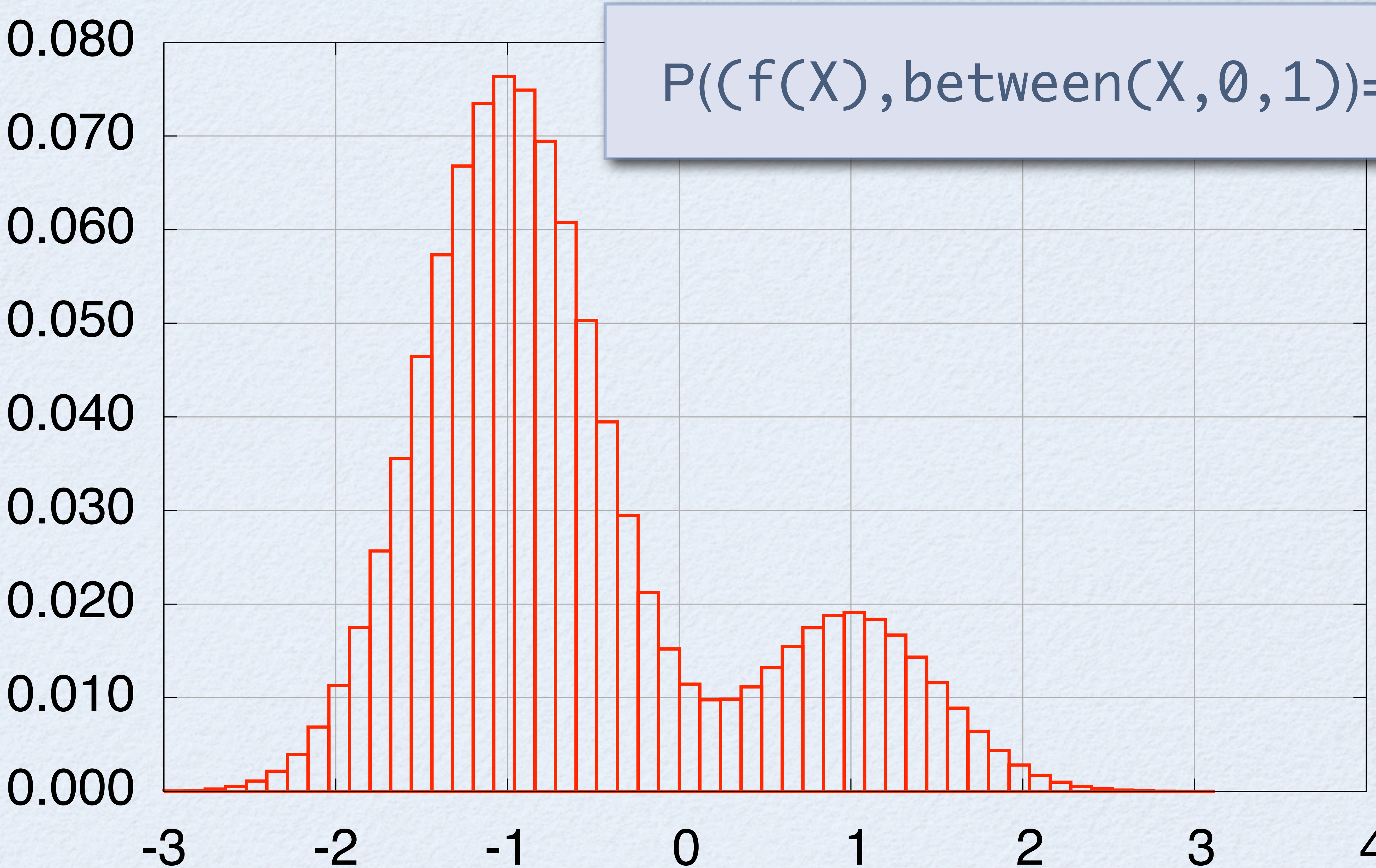


40 Intervals



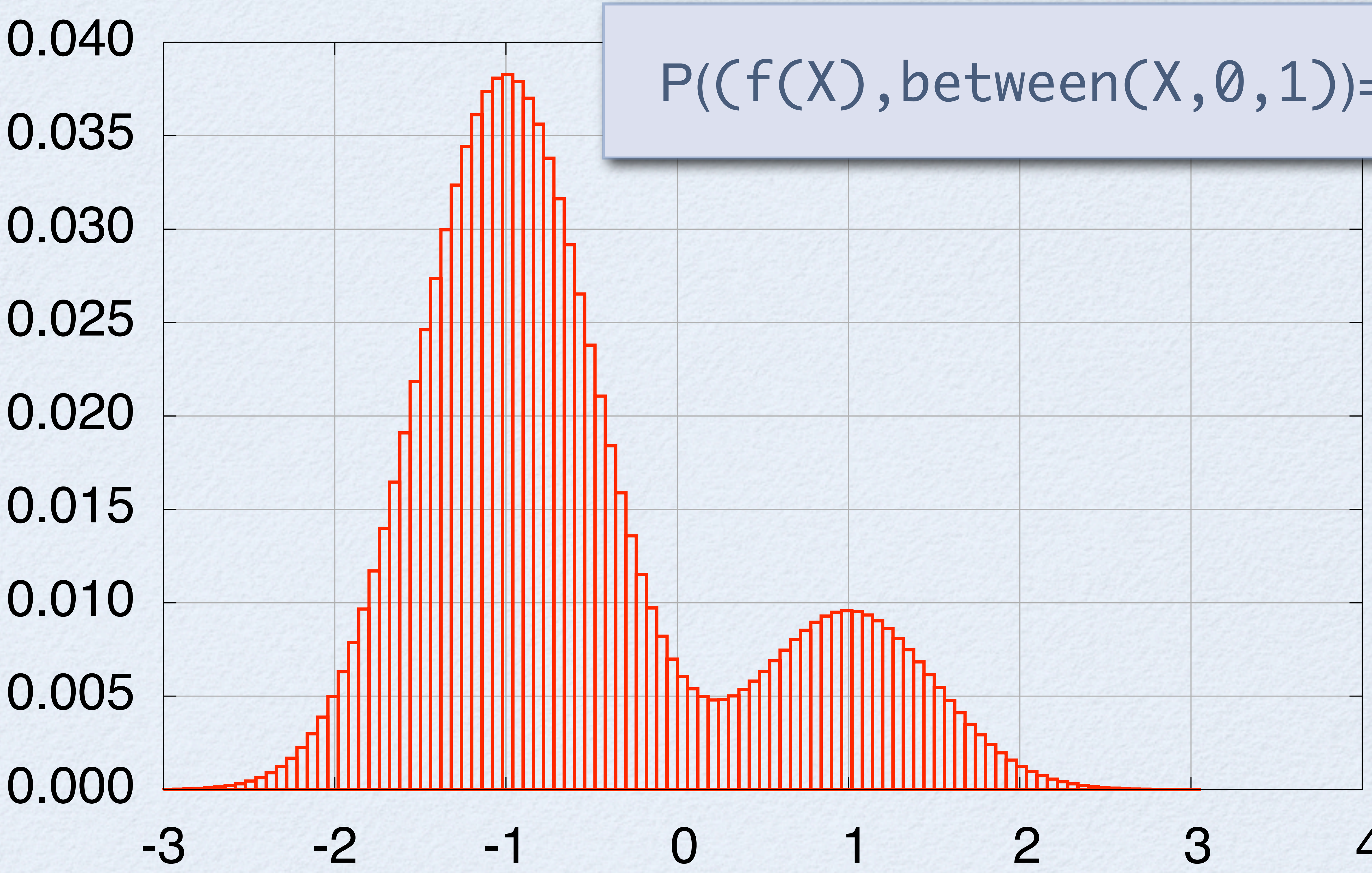
P((f(X),between(X,0,1))=?

50 Intervals



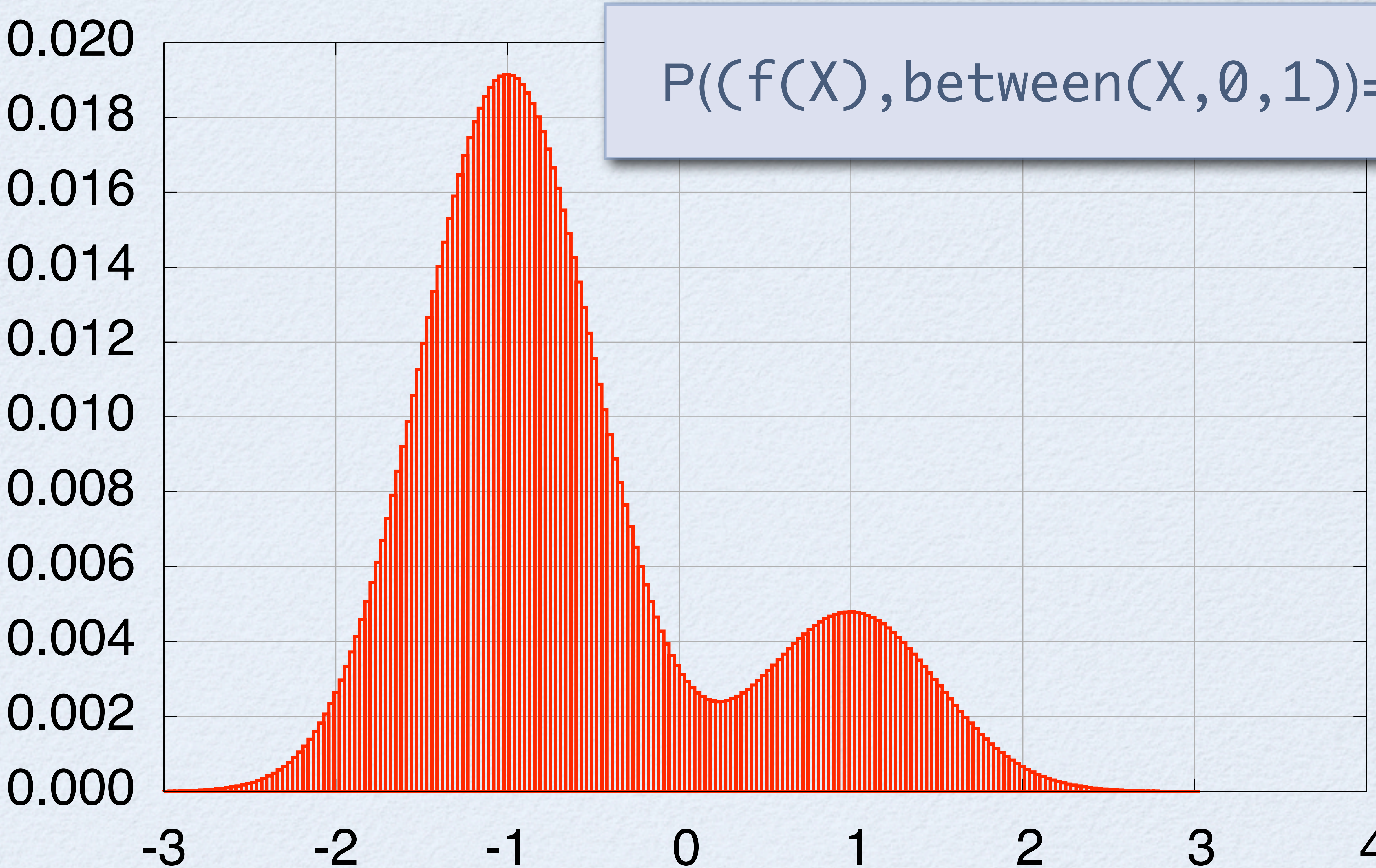

```
P((f(X),between(X,0,1))=?
```

100 Intervals



P((f(X),between(X,0,1))=?

200 Intervals



P((f(X),between(X,0,1))=?

Outline

- Introduction
- Hybrid Problog
- **Inference**
- Learning
- Increase Expressivity

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.

in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.

in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

room_has_window :-
  in_office,
  office_has_window.
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
office_has_window
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
office_has_window
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

room_has_window :-
  in_office,
  office_has_window.
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
office_has_window
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
office_has_window
in_corridor
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
office_has_window
in_corridor
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Sampling

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

```
width(2.3)
length(7.9)
office_has_window
in_corridor
```

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Exact Inference

- Step 1 run SLD resolution and collect all proofs
- Step 2 disjoin proofs
- Step 3 Build DNF
- Step 4 make intervals mutually exclusive
- Step 5 Build BDD

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{}	{}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{}	{}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts

{}

Cont. Facts

{}

Domains

{}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{}	{}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{}	{}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{width(W)}	{W in ℝ}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{width(W),length(L)}	{W in IR, L in IR}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

room_has_window :-
  in_office,
  office_has_window.
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{width(W),length(L)}	{W in [2,4], L in lR}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

room_has_window :-
  in_office,
  office_has_window.
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{}	{width(W),length(L)}	{W in [2,4], L in [2,4]}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{office_has_window}	{width(W),length(L)}	{W in [2,4], L in [2,4]}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).
```

```
in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).
```

```
room_has_window :-
  in_office,
  office_has_window.
```

```
room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts

{office_has_window}

Cont. Facts

{width(W),length(L)}

Domains

{W in [2,4], L in [2,4]}

SLD Resolution

```
(W,gaussian(2,1)) :: width(W).
(L,gaussian(9,3)) :: length(L).
0.8 :: office_has_window.
0.001 :: corridor_has_window.
```

P(room_has_window)=?

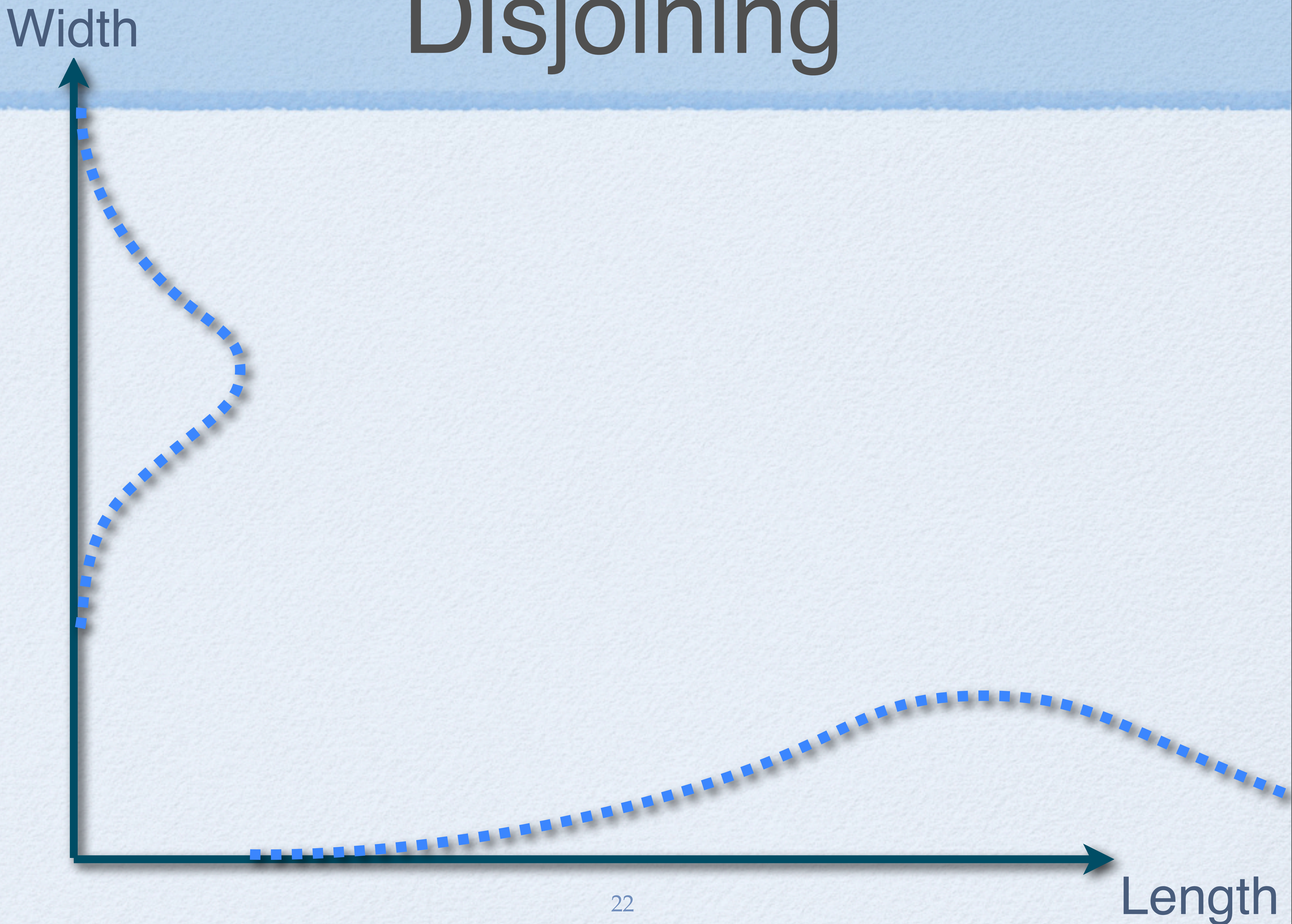
```
in_office :-
  width(W), length(L),
  in_interval(W,2,4),
  in_interval(L,2,4).

in_corridor :-
  width(W), length(L),
  below(W,2.5),
  above(L,3).

room_has_window :-
  in_office,
  office_has_window.

room_has_window :-
  in_corridor,
  corridor_has_window.
```

Prob. Facts	Cont. Facts	Domains
{office_has_window}	{width(W),length(L)}	{W in [2,4], L in [2,4]}
{corridor_has_window}	{width(W),length(L)}	{W in (-oo,2.5), L in (3,oo)}

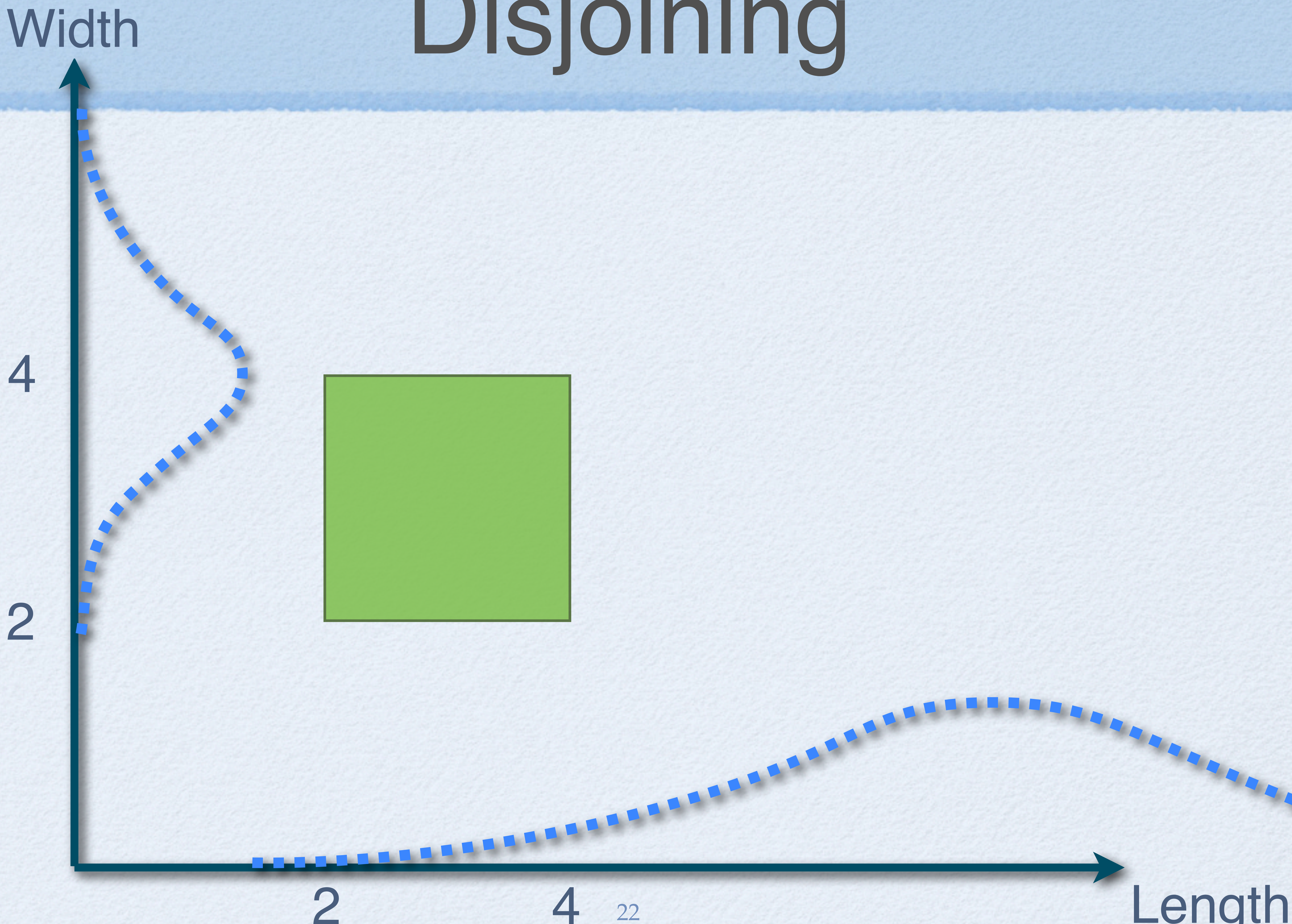
Disjoining

- Proofs overlap
  - with respect to probabilistic facts
  - with respect to continuous facts
- Naively adding probabilities yields wrong values
- disjoint sum problem
- extend ProbLog’s BDD algorithm for continuous distributions

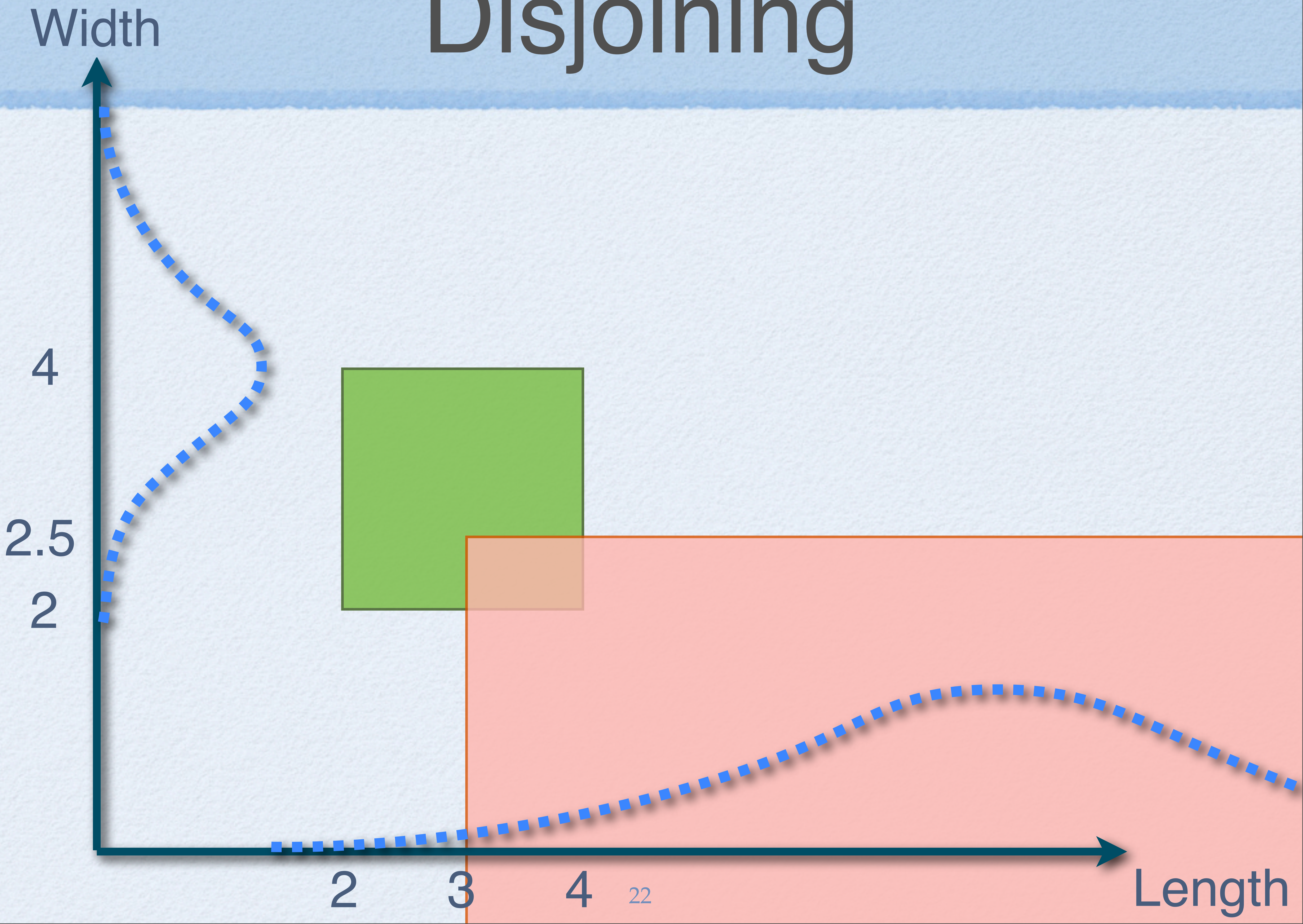
Disjoining



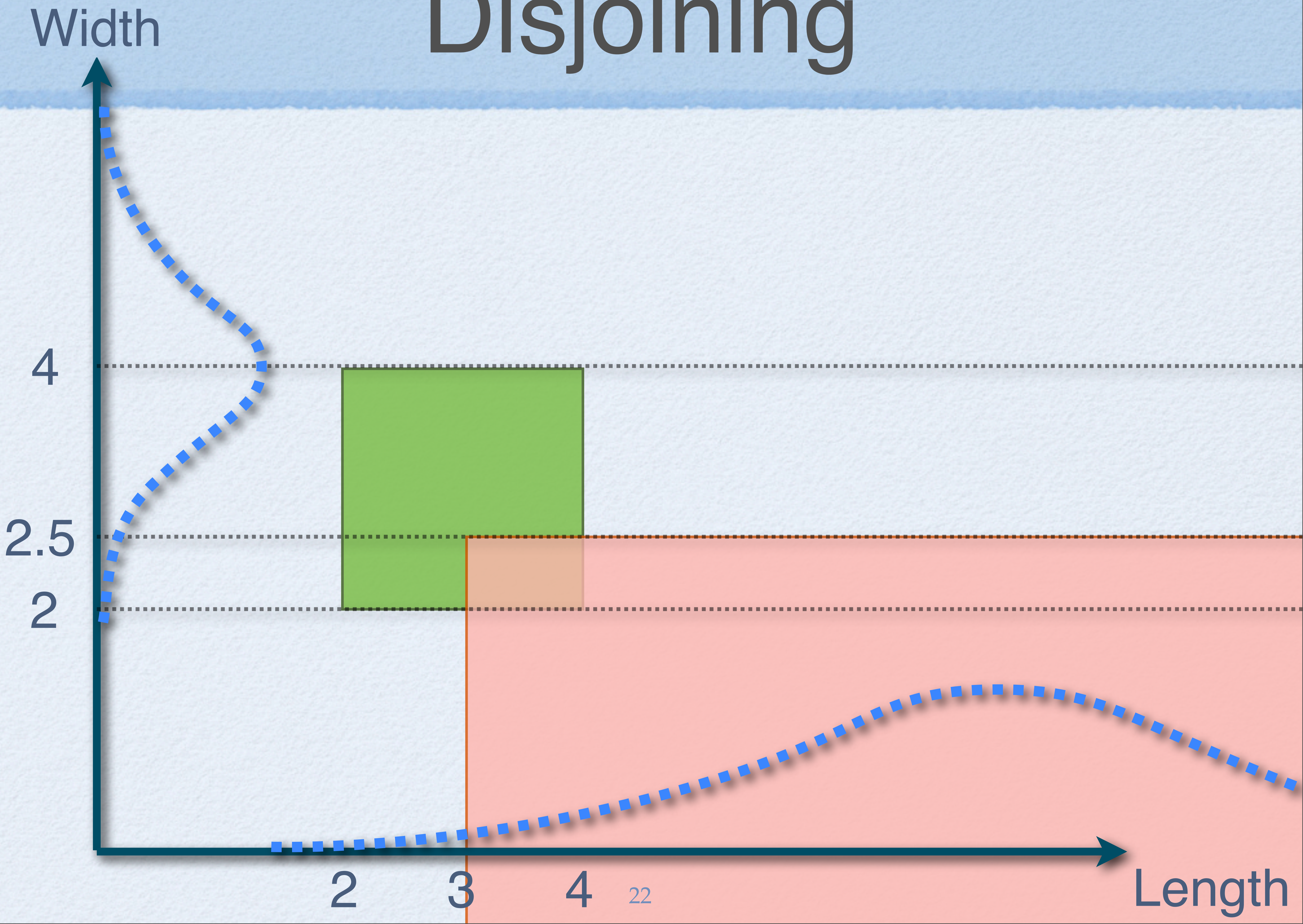
Disjoining



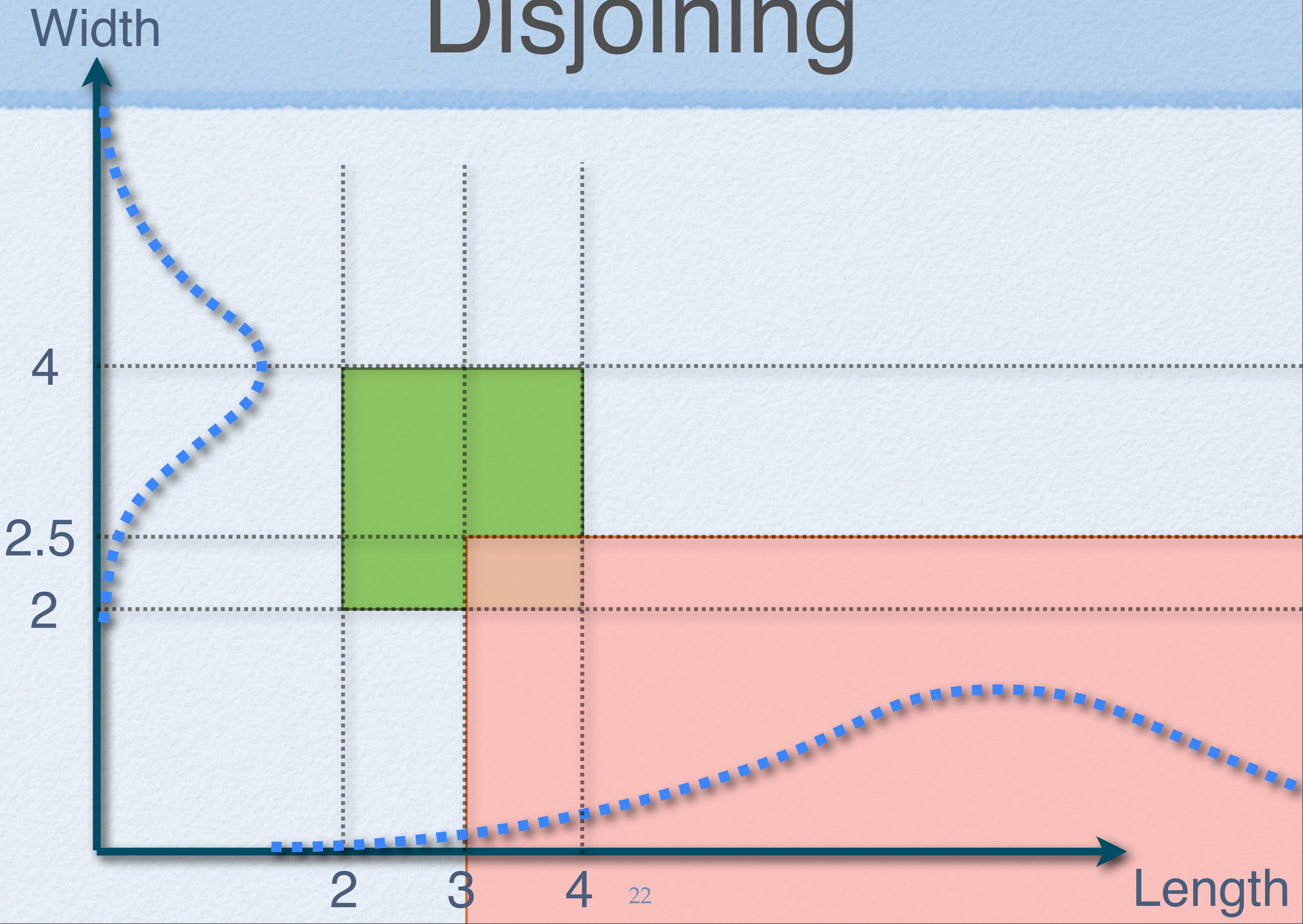
Disjoining



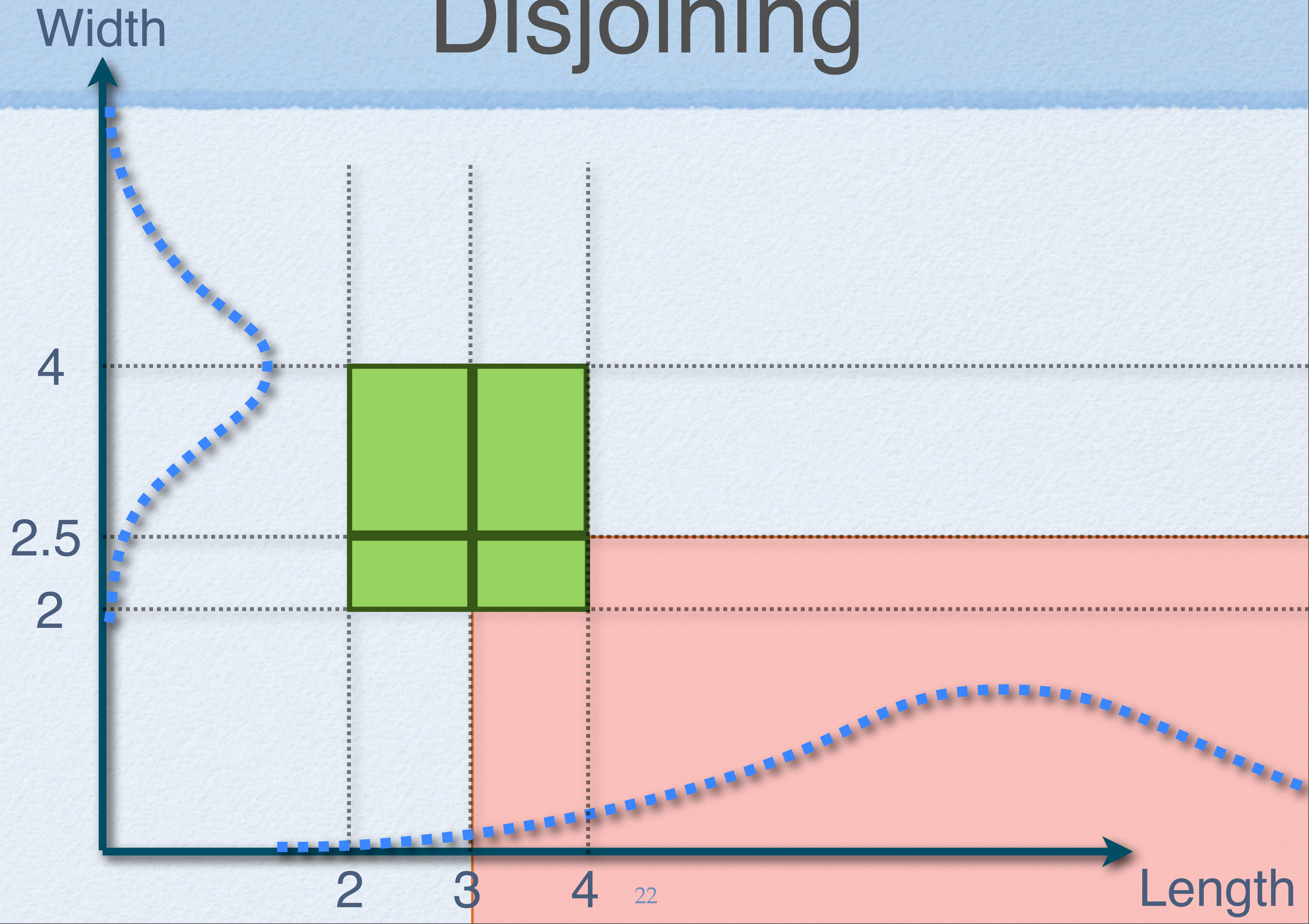
Disjoining



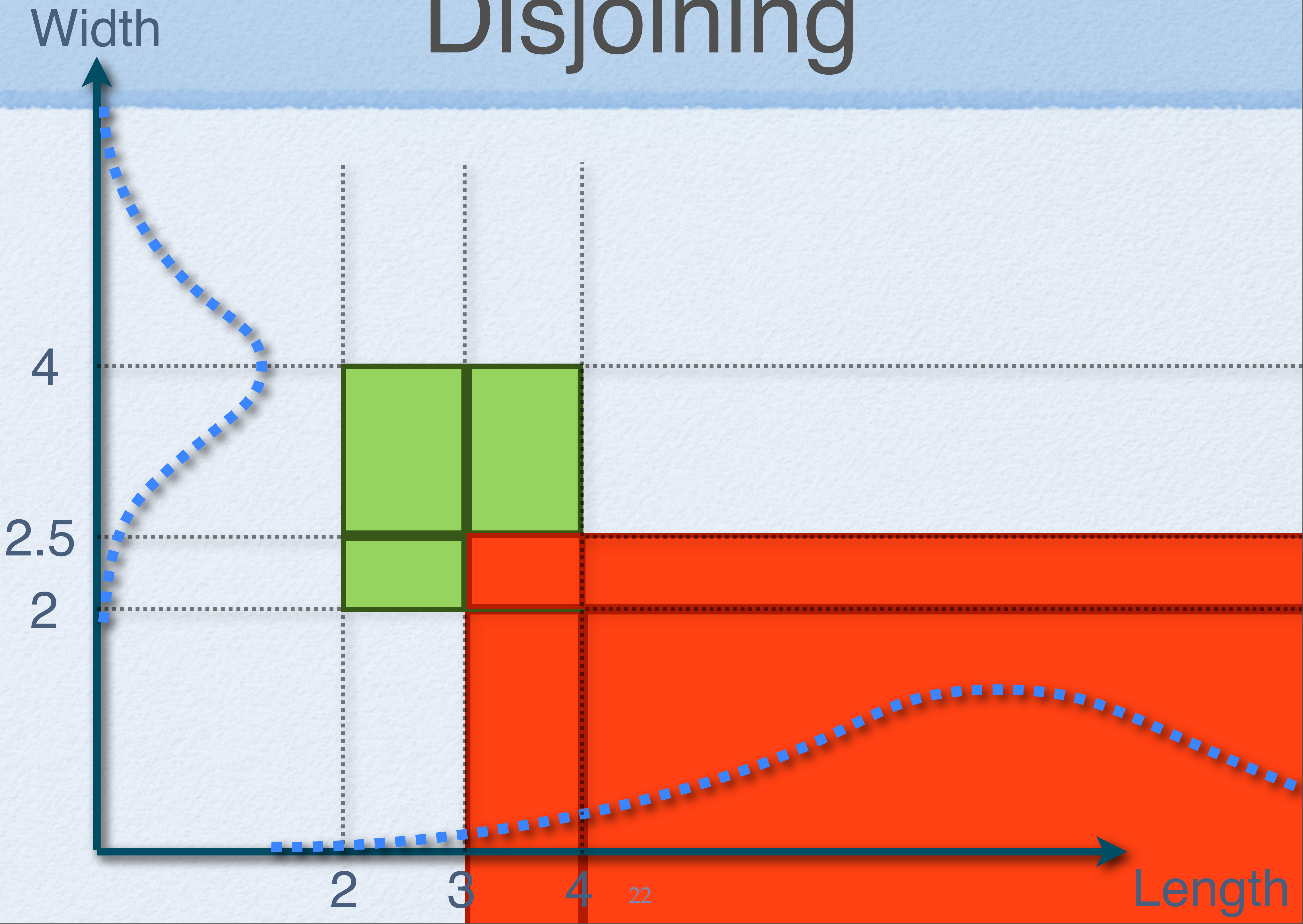
Disjoining



Disjoining



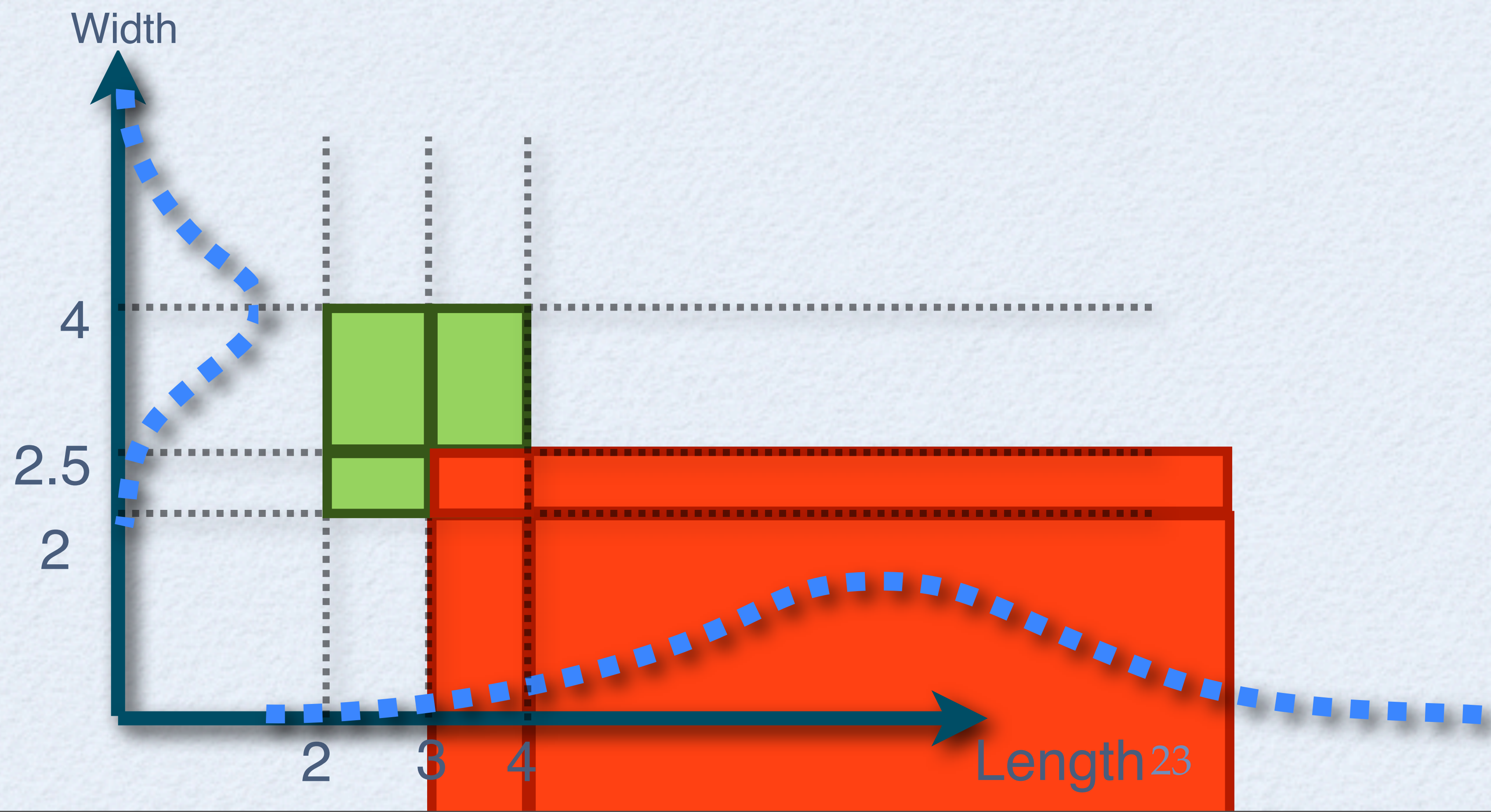
Disjoining



Disjoining (2)

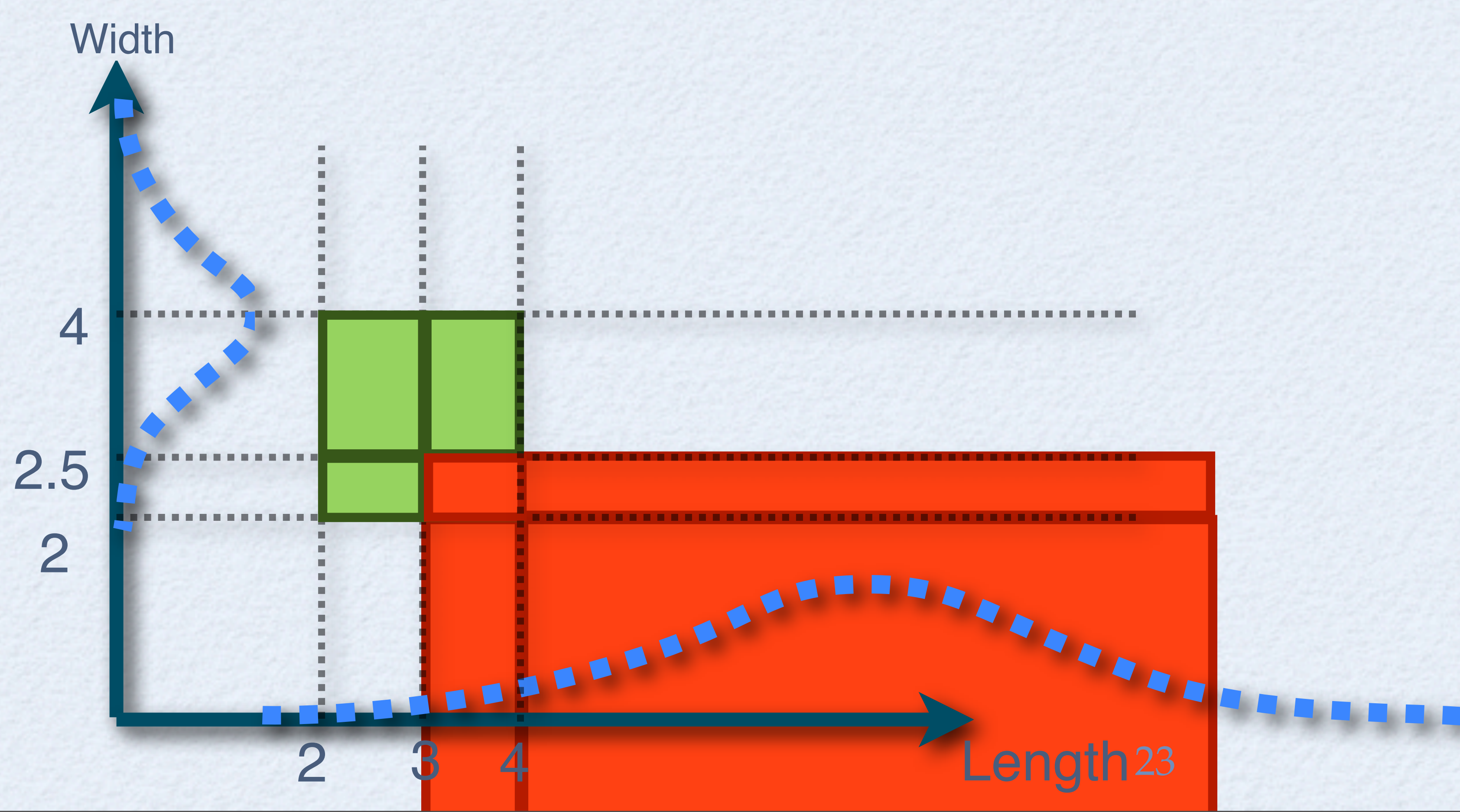
{office_has_window}   {width(W),length(L)} {W ∈ [2,4],L ∈ [2,4]}

{corridor_has_window} {width(W),length(L)} {W ∈ (-oo,2.5),L ∈ (3,oo)}



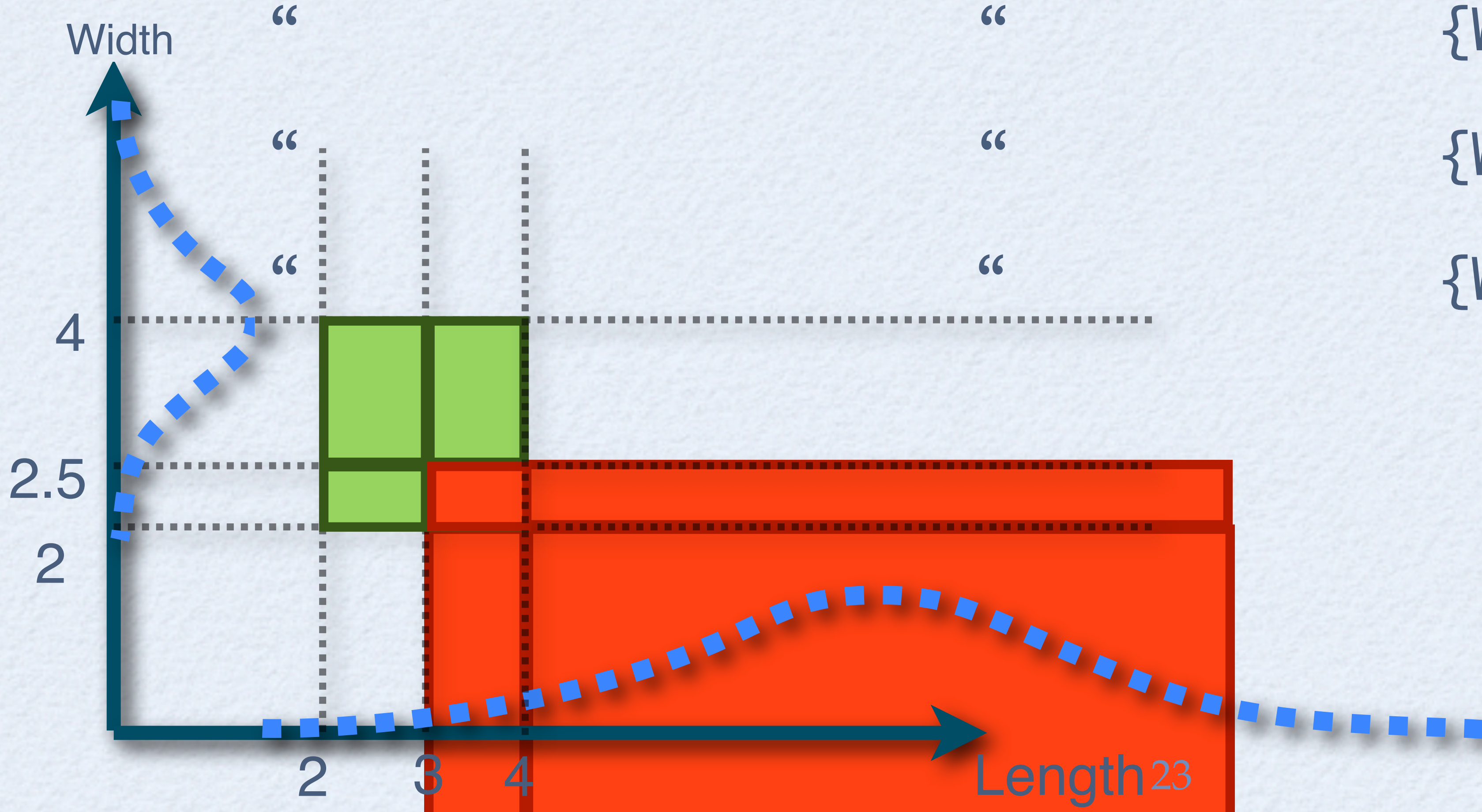
Disjoining (2)

{office_has_window}	{width(W),length(L)}	{W ∈ [2,2.5),L ∈ [2,3)}
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2.5),L ∈ (3,oo)}



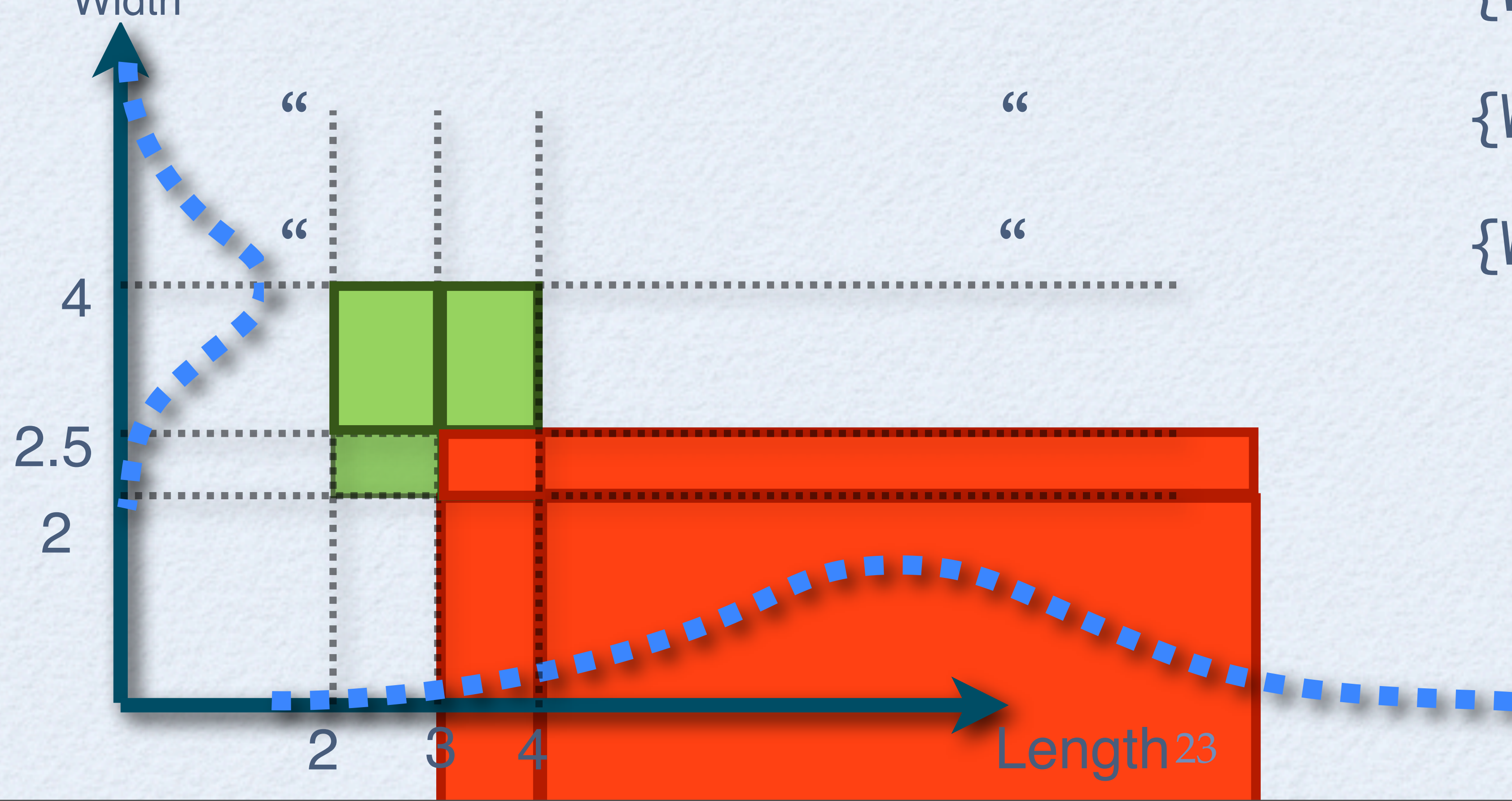
Disjoining (2)

{office_has_window}	{width(W),length(L)}	{W ∈ [2,2.5),L ∈ [2,3)}
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}



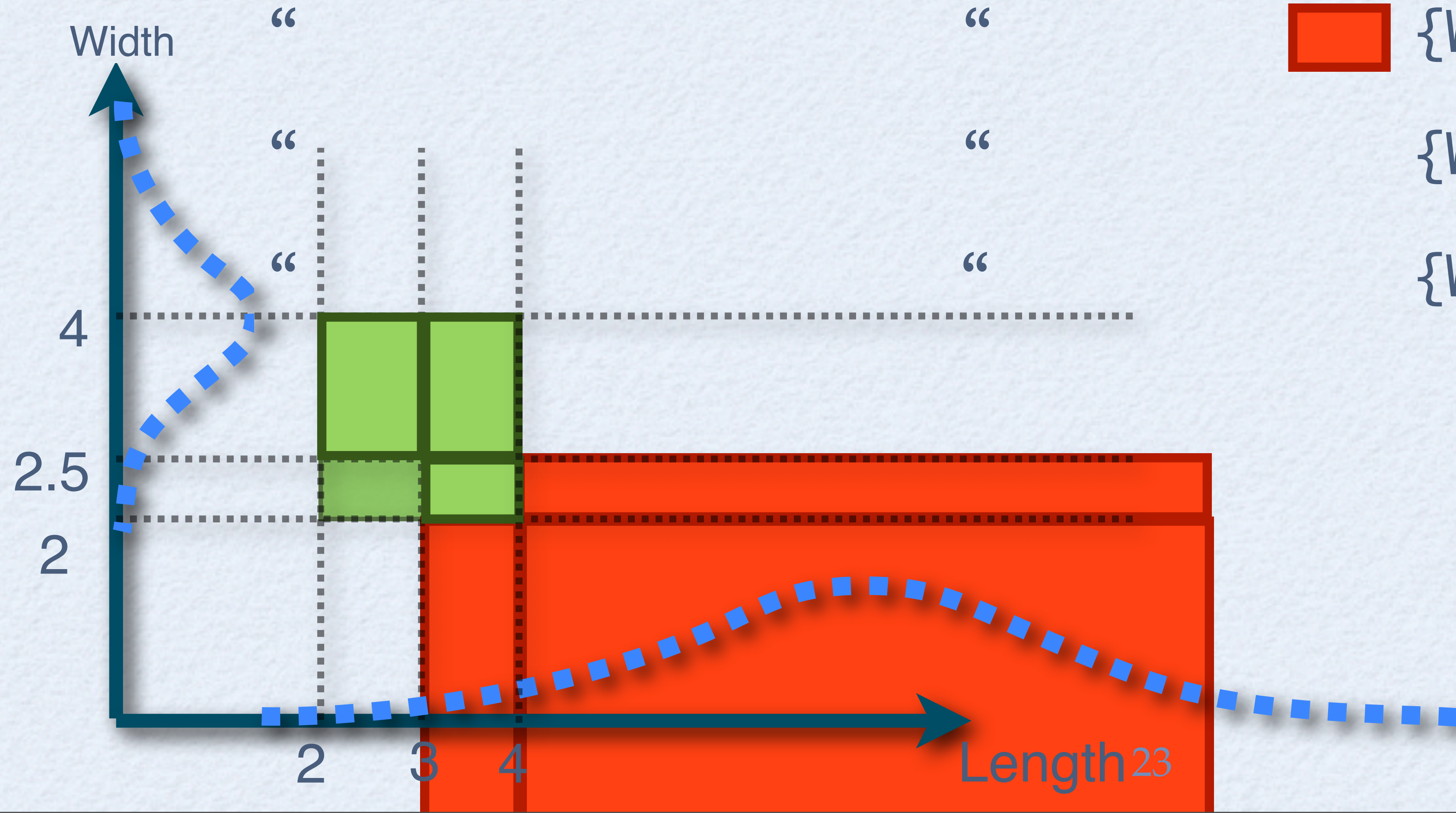
Disjoining (2)

{office_has_window}	{width(W),length(	{W ∈ [2,2.5),L ∈ [2,3)}
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}



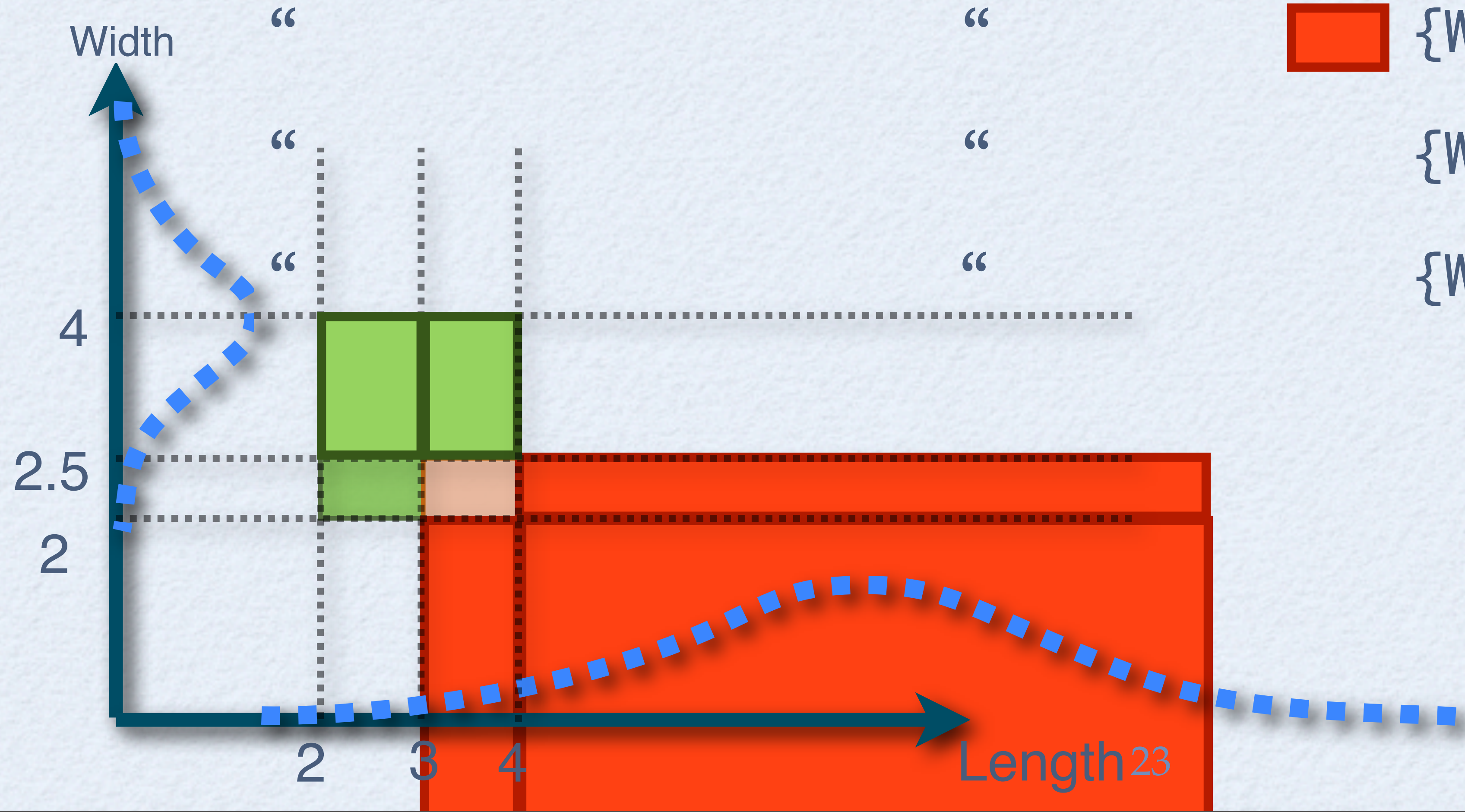
Disjoining (2)

{office_has_window}	{width(W),length(	{W ∈ [2,2.5),L ∈ [2,3)}
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}



Disjoining (2)

{office_has_window}	{width(W),length(	{W ∈ [2,2.5),L ∈ [2,3)}
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}



Disjoining (2)

{office_has_window} {width(W),length({W ∈ [2,2.5),L ∈ [2,3)}

“ “ {W ∈ [2.5,4],L ∈ [2,3)}

“ “ {W ∈ [2,2.5),L ∈ [3,4]}

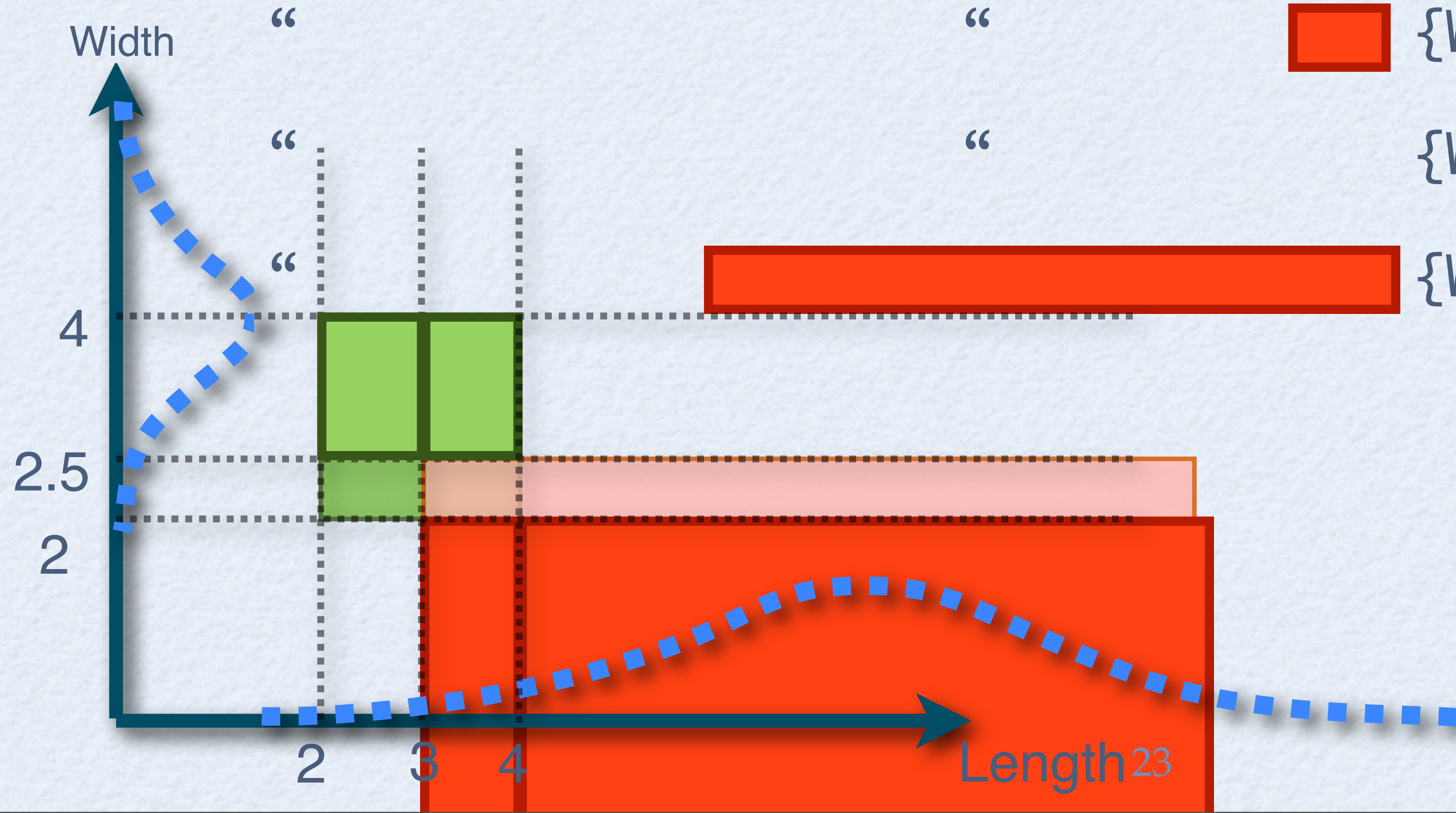
“ “ {W ∈ [2.5,4],L ∈ [3,4]}

{corridor_has_window} {width(W),length(L)} {W ∈ (-oo,2), L ∈ (3,4)}

“ “ {W ∈ [2,2.5),L ∈ (3,4)}

“ “ {W ∈ (-oo,2),L ∈ [4,oo)}

“ {W ∈ [2,2.5),L ∈ [4,oo)}



Building the DNF

{office_has_window}	{width(W),length(L)}	{W ∈ [2,2.5),L ∈ [2,3)}
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}

Building the DNF

office_has_window ^	width(W)$_{W \in [2,2.5)}$ ^	length(L)$_{L \in [2,3)}$
“	“	{W ∈ [2.5,4],L ∈ [2,3)}
“	“	{W ∈ [2,2.5),L ∈ [3,4]}
“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}

Building the DNF

office_has_window ^ width(W)$_{W \in [2,2.5)}$ ^ length(L)$_{L \in [2,3)}$

office_has_window ^ width(W)$_{W \in [2.5,4]}$ ^ length(L)$_{L \in [2,3)}$

"	"	{W ∈ [2,2.5),L ∈ [3,4]}
"	"	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
"	"	{W ∈ [2,2.5),L ∈ (3,4)}
"	"	{W ∈ (-oo,2),L ∈ [4,oo)}
"	"	{W ∈ [2,2.5),L ∈ [4,oo)}

Building the DNF

office_has_window ^ width(W)$_{W\in[2,2.5)}$ ^ length(L)$_{L\in[2,3)}$

office_has_window ^ width(W)$_{W\in[2.5,4]}$ ^ length(L)$_{L\in[2,3)}$

office_has_window ^ width(W)$_{W\in[2,2.5)}$ ^ length(L)$_{L\in[3,4]}$

“	“	{W ∈ [2.5,4],L ∈ [3,4]}
{corridor_has_window}	{width(W),length(L)}	{W ∈ (-oo,2), L ∈ (3,4)}
“	“	{W ∈ [2,2.5),L ∈ (3,4)}
“	“	{W ∈ (-oo,2),L ∈ [4,oo)}
“	“	{W ∈ [2,2.5),L ∈ [4,oo)}

Building the DNF

$office\_has\_window \wedge width(W)_{W \in [2,2.5)} \wedge length(L)_{L \in [2,3)}$

$office\_has\_window \wedge width(W)_{W \in [2.5,4]} \wedge length(L)_{L \in [2,3)}$

$office\_has\_window \wedge width(W)_{W \in [2,2.5)} \wedge length(L)_{L \in [3,4]}$

$office\_has\_window \wedge width(W)_{W \in [2.5,4]} \wedge length(L)_{L \in [3,4]}$

{corridor_has_window}	{width(W),length(L)}	$\{W \in (-oo,2),\ L \in (3,4)\}$
“	“	$\{W \in [2,2.5), L \in (3,4)\}$
“	“	$\{W \in (-oo,2), L \in [4,oo)\}$
“	“	$\{W \in [2,2.5), L \in [4,oo)\}$

Building the DNF

office_has_window ∧ width(W)$_{W\in[2,2.5)}$ ∧ length(L)$_{L\in[2,3)}$

office_has_window ∧ width(W)$_{W\in[2.5,4]}$ ∧ length(L)$_{L\in[2,3)}$

office_has_window ∧ width(W)$_{W\in[2,2.5)}$ ∧ length(L)$_{L\in[3,4]}$

office_has_window ∧ width(W)$_{W\in[2.5,4]}$ ∧ length(L)$_{L\in[3,4]}$

corridor_has_window ∧ width(W)$_{W\in(-oo,2)}$ ∧ length(L)$_{L\in(3,4)}$

corridor_has_window ∧ width(W)$_{W\in[2,2.5)}$ ∧ length(L)$_{L\in(3,4)}$

corridor_has_window ∧ width(W)$_{W\in[-oo,2)}$ ∧ length(L)$_{L\in[4,oo)}$

corridor_has_window ∧ width(W)$_{W\in[2,2.5)}$ ∧ length(L)$_{L\in[4,oo)}$

Enforcing Exclusiveness

$\texttt{width(W)}_{W \in [2,2.5)}$ vs. $\texttt{width(W)}_{W \in [2.5,4]}$

Enforcing Exclusiveness

$width(W)_{W\in[2,2.5)}$ vs. $width(W)_{W\in[2.5,4]}$

Replace **by**

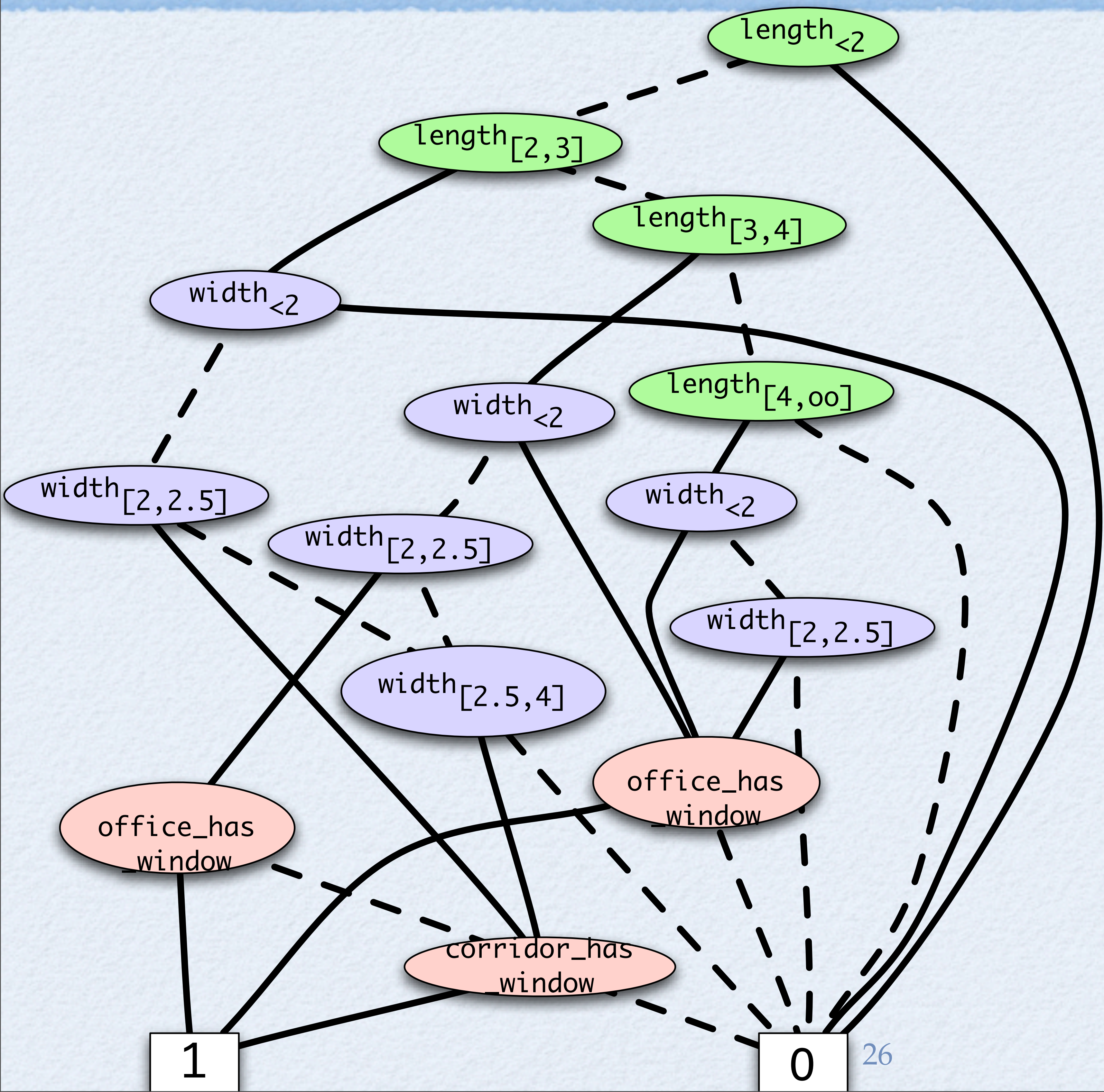
$width(W)_{W\in(-oo,2)}$ → $width(W)_{W\in(-oo,2)}$

$width(W)_{W\in[2,2.5)}$ → $\neg width(W)_{W\in(-oo,2)} \wedge width(W)_{W\in[2,2.5)}$

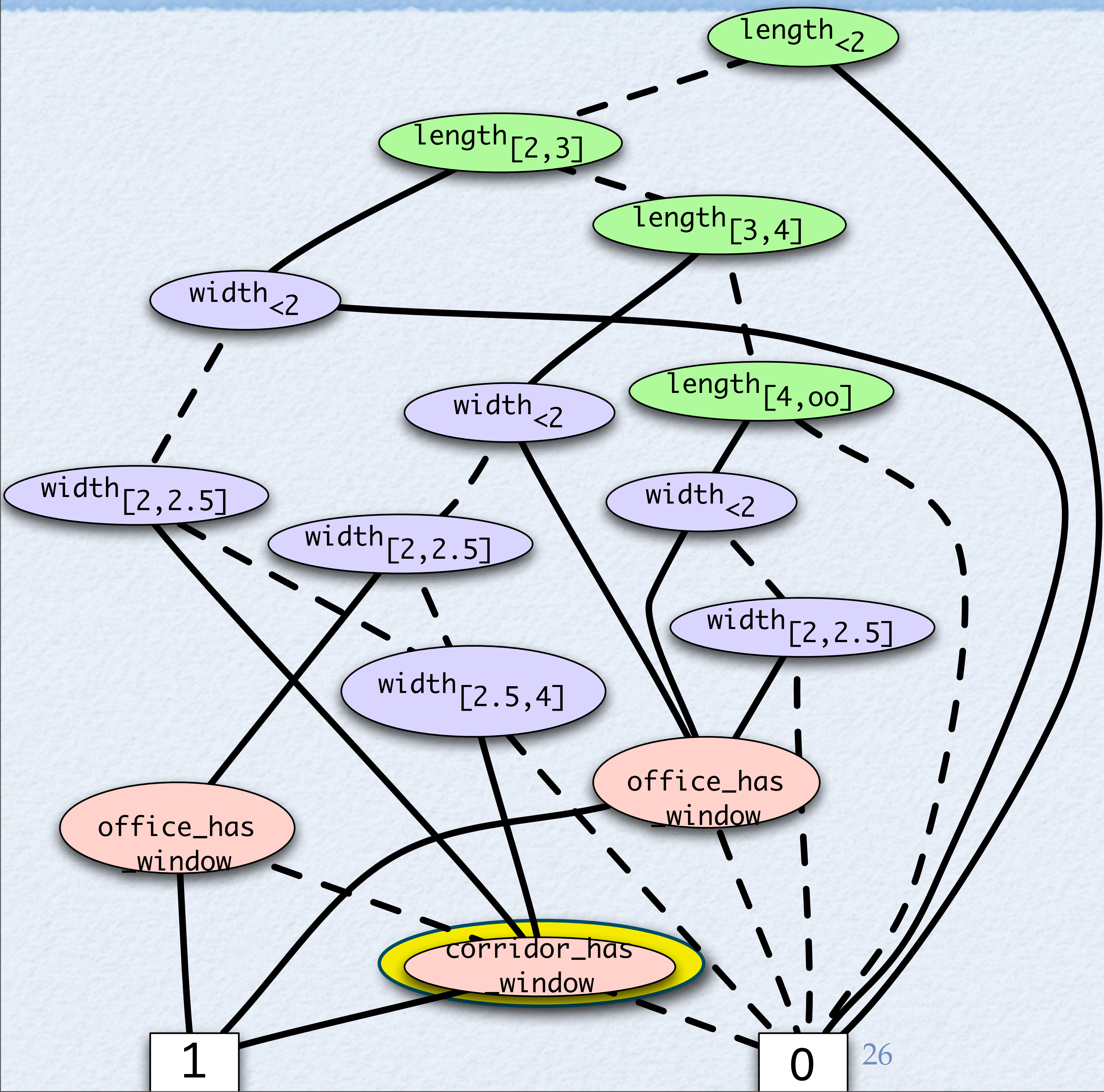
$width(W)_{W\in[2.5,4]}$ → $\neg width(W)_{W\in(-oo,2)} \wedge \neg width(W)_{W\in[2,2.5)} \wedge width(W)_{W\in[2.5,4]}$

$width(W)_{W\in(4,oo)}$ → $\neg width(W)_{W\in(-oo,2)} \wedge \neg width(W)_{W\in[2,2.5)} \wedge \neg width(W)_{W\in[2.5,4]} \wedge width(W)_{W\in(4,oo)}$

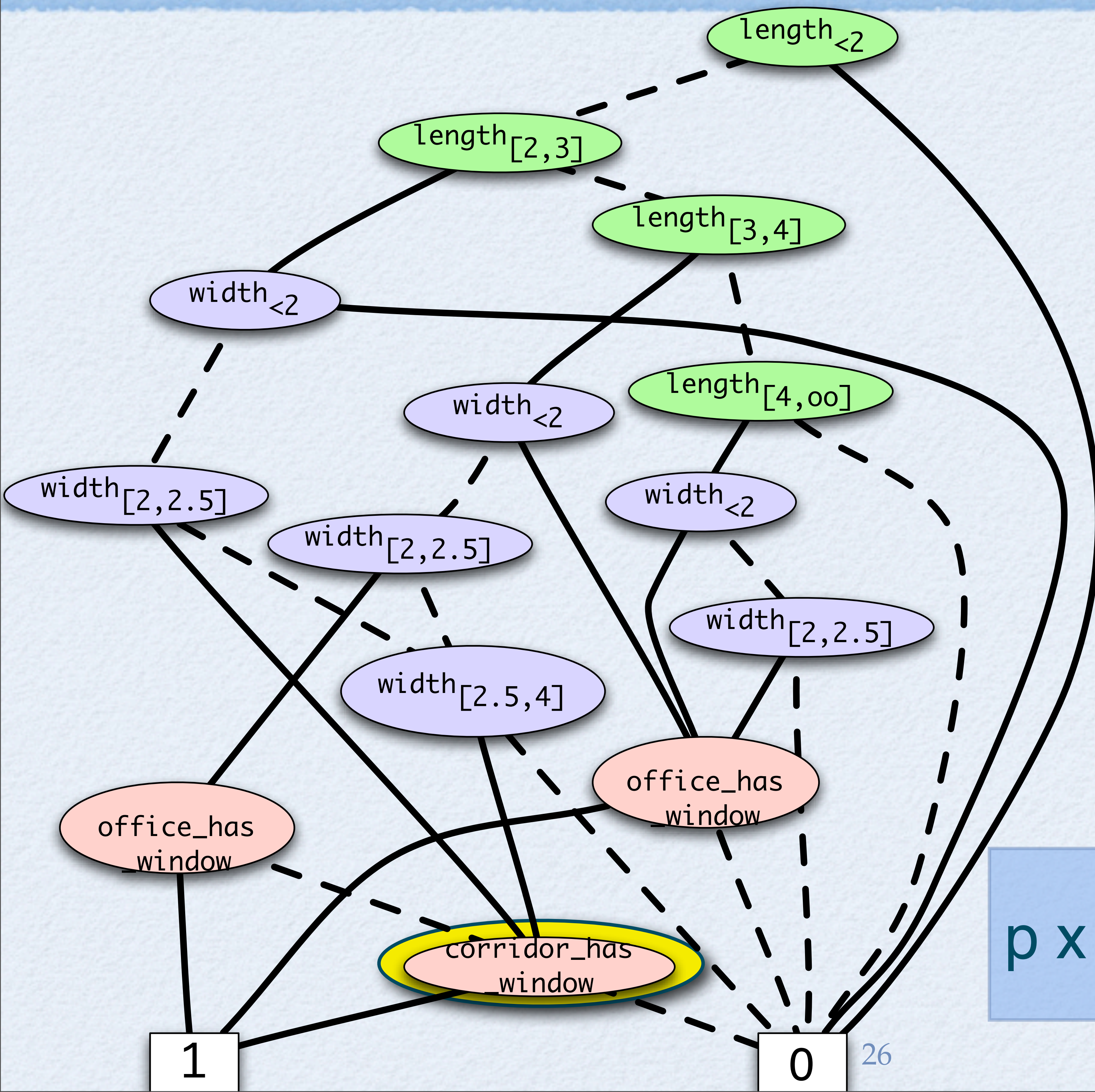
Build & Evaluate BDD



Build & Evaluate BDD

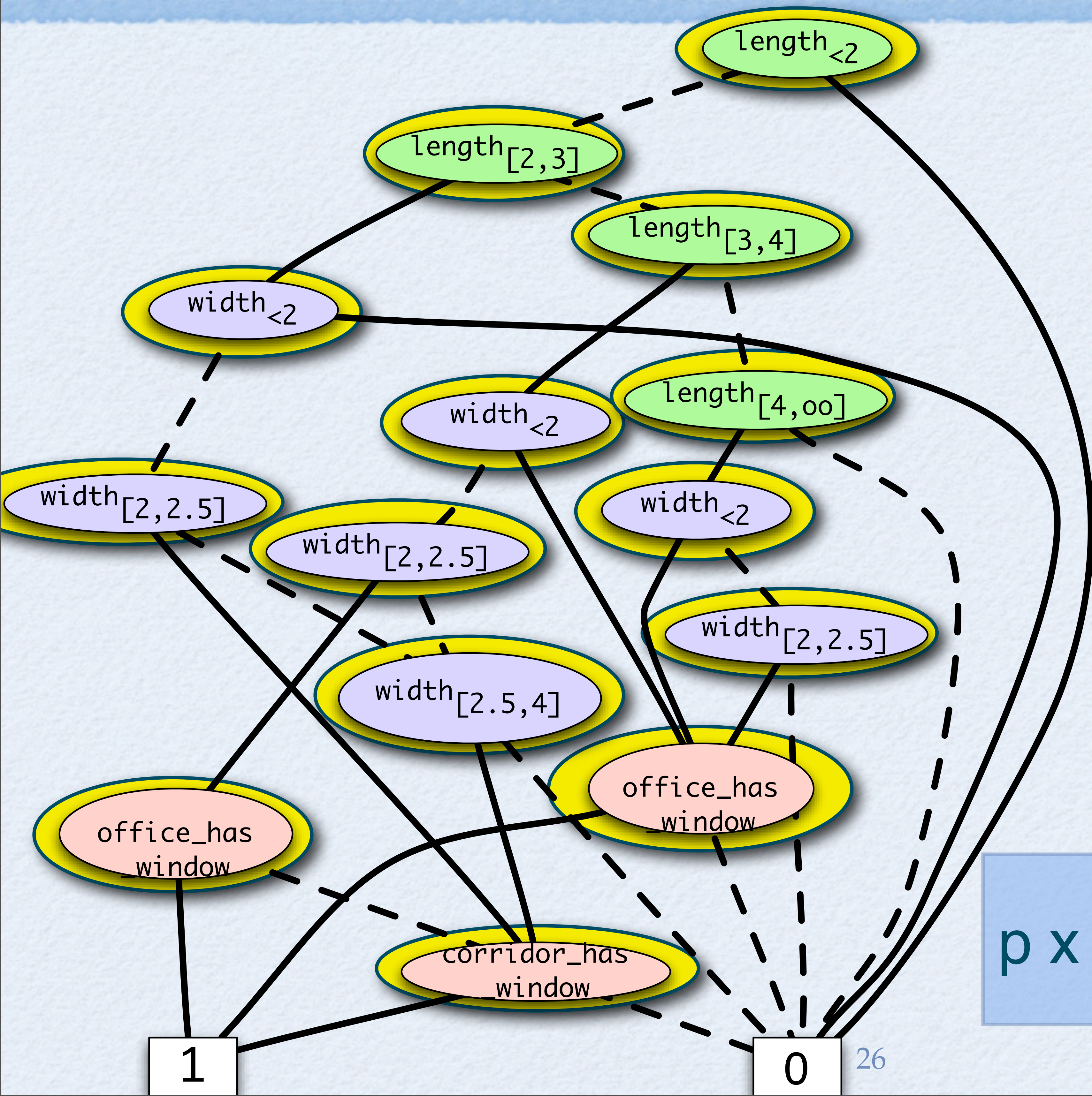


Build & Evaluate BDD



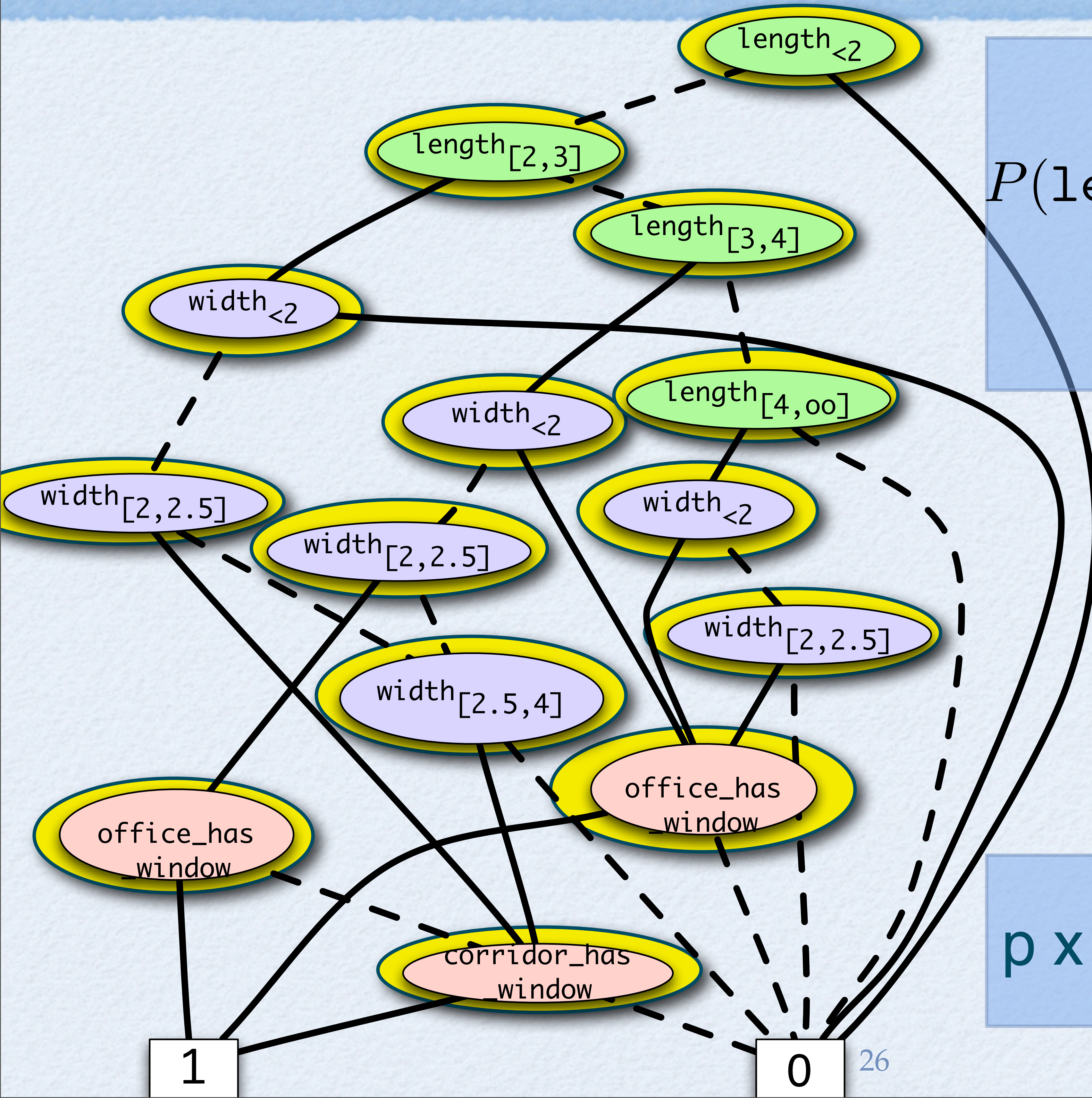
p x p_high + (1-p) x p_low

Build & Evaluate BDD



p x p_high + (1-p) x p_low

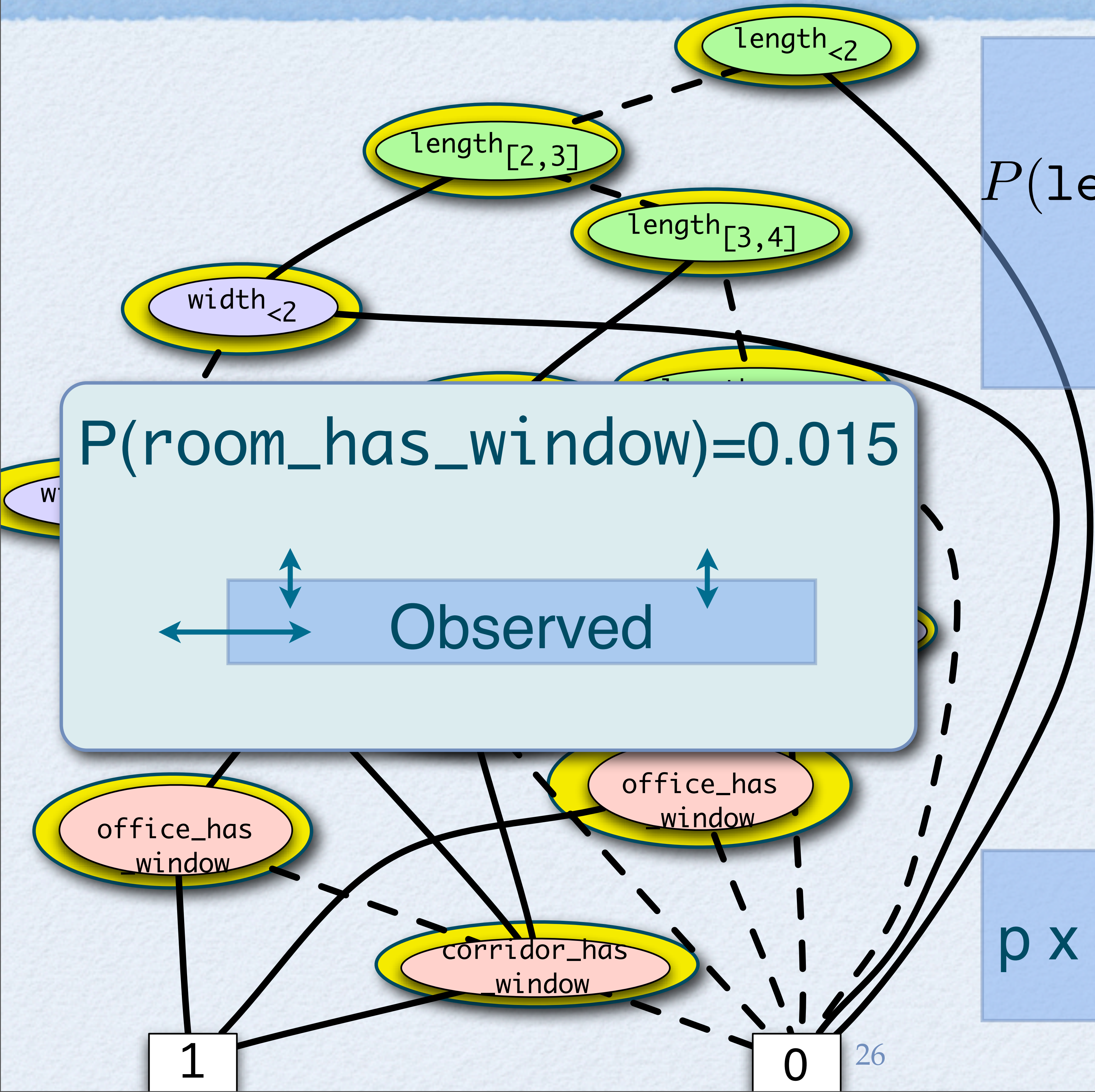
Build & Evaluate BDD



$$P(\texttt{length}_{[2,3]}) = \frac{\int_{2}^{3} \mathcal{N}(x)dx}{\int_{-\infty}^{2} \mathcal{N}(x)dx}$$

p x p_high + (1-p) x p_low

Build & Evaluate BDD



P(room_has_window)=0.015

Observed

$$P(\texttt{length}_{[2,3]}) = \frac{\int_2^3 \mathcal{N}(x)dx}{\int_{-\infty}^2 \mathcal{N}(x)dx}$$

p x p_high + (1-p) x p_low

Outline

- Introduction
- Hybrid Problog
- Inference
- **Learning**
- Increase Expressivity

What to Learn?

- fact probabilities
  ?? :: office_has_window
- parameters of the the continuous distributions
  (X, gaussian(??,??)) :: length(X)
- distributions
  (X,D)::length(X)
- decision boundaries
  ..., length(X), above(X,??), ...

Parameter Learning

[Gutmann et al. 2008]

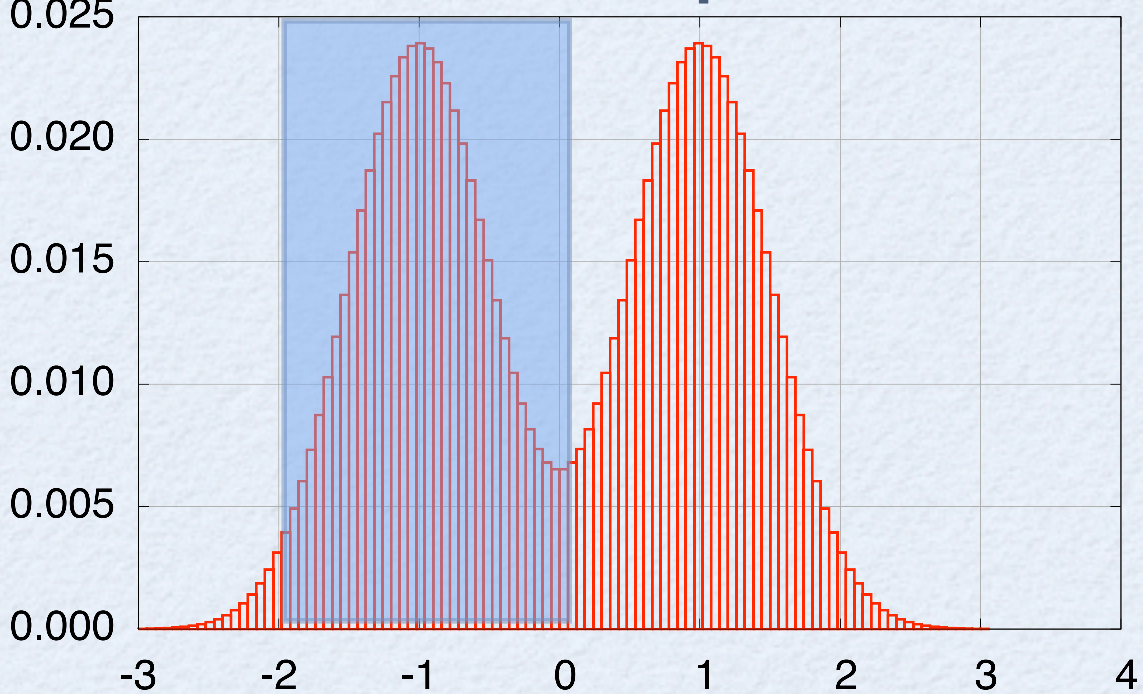
Can be calculated	Goal:	Training Data
$P_s(q_1)$	=	$q_1 = 0.32$
$P_s(q_2)$	=	$q_2 = 0.23$
$P_s(q_3)$	=	$q_3 = 0.90$
...		...

$$MSE(T) = \frac{1}{K} \sum_{1 \leq i \leq K} \left(P_s(q_i|T) - \tilde{p}_i \right)^2$$

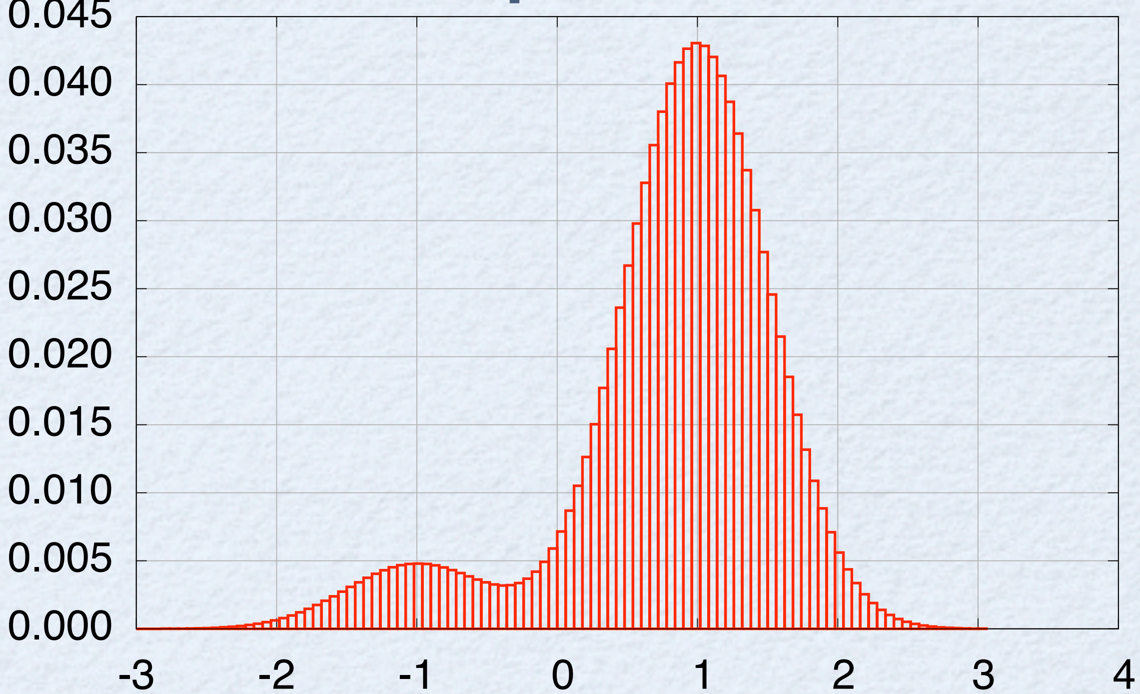
Parameter Learning

- goal: learn the mixture parameter from training examples of the form P(low <= X <= high)=p
- P((f(X),between(X,-2,0))=0.12

flip=0.5



flip=0.1



Increase Expressivity

Increase Expressivity

- allow functions of one variable

```
(X, gaussian(0,1)) :: f(X)
g :- f(X), X2 is X*X, below(X2, 16).
h :- f(X), Y is sin(X), below(X,0).
```

Increase Expressivity

- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(X), Y is sin(X), below(X,0).
```

Increase Expressivity

- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(X), Y is sin(X), below(X,0).
```

- allow multivariate distributions

```
((X,Y), density) :: f(X,Y)
```

Increase Expressivity

- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(X), Y is sin(X), below(X,0).
```

- allow multivariate distributions

n-dimensional integral must be computable

Increase Expressivity

- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(X), Y is sin(X), below(X,0).
```

- allow multivariate distributions

n-dimensional integral must be computable

- allow comparing two continuous variables

```
(X, gaussian(0,1)) :: f(X).
(X, gaussian(5,3)) :: g(X).
h :- f(X), g(Y),above(X,Y).
```

Increase Expressivity

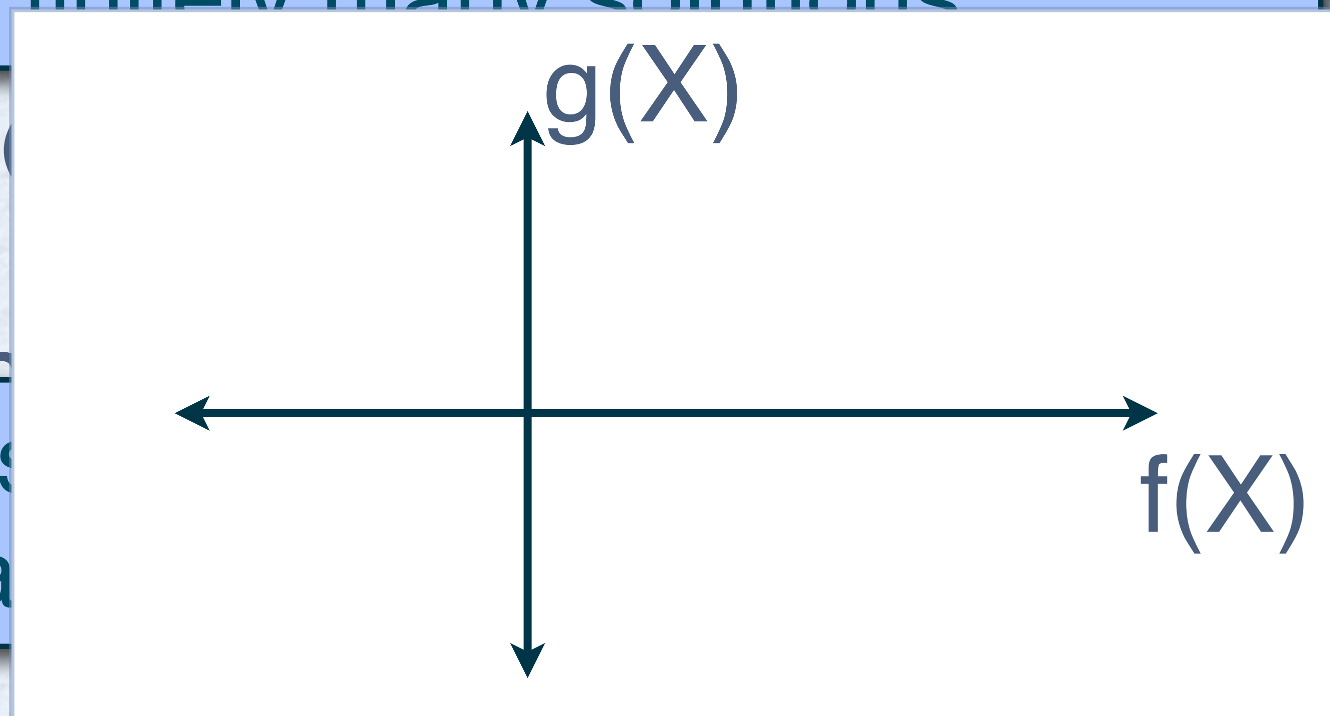
- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(               (X,0).
```

- allow m

n-dimens
computa



- allow comparing two continuous variables

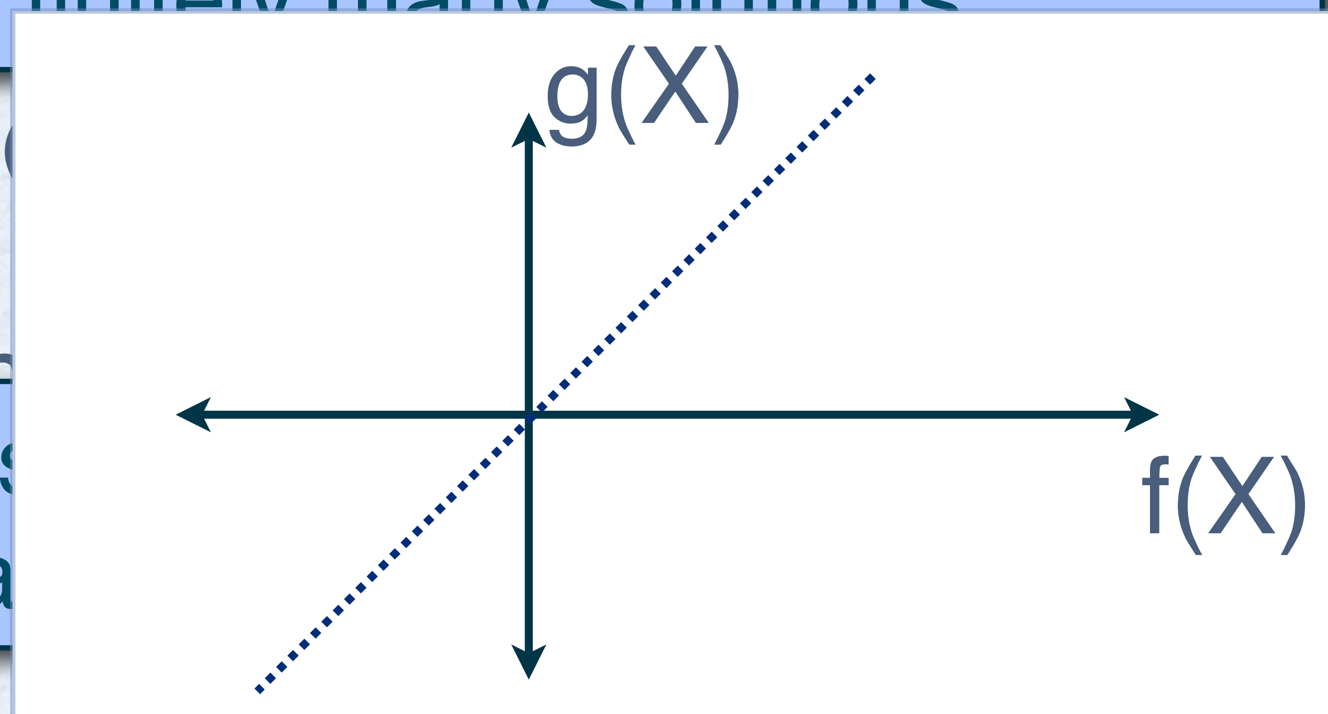
```
(X, gaussian(0,1)) :: f(X).
(X, gaussian(5,3)) :: g(X).
h :- f(X), g(Y),above(X,Y).
```

Increase Expressivity

- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(                                  (X,0).
```



- allow m

n-dimens
computa

- allow comparing two continuous variables

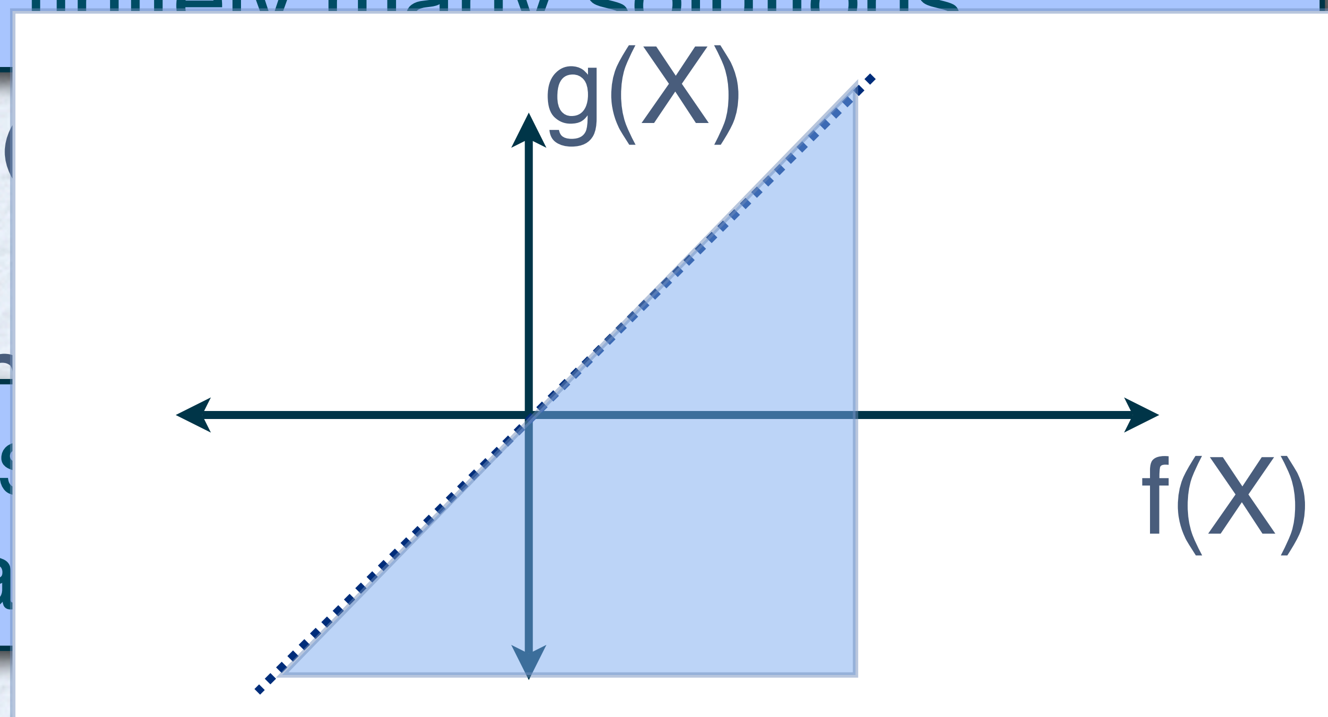
```
(X, gaussian(0,1)) :: f(X).
(X, gaussian(5,3)) :: g(X).
h :- f(X), g(Y),above(X,Y).
```

Increase Expressivity

- allow functions of one variable

one needs to solve f(X)=0
requires finitely many solutions

```
2, 16).
h :- f(                    (X,0).
```



- allow m

n-dimens
computa

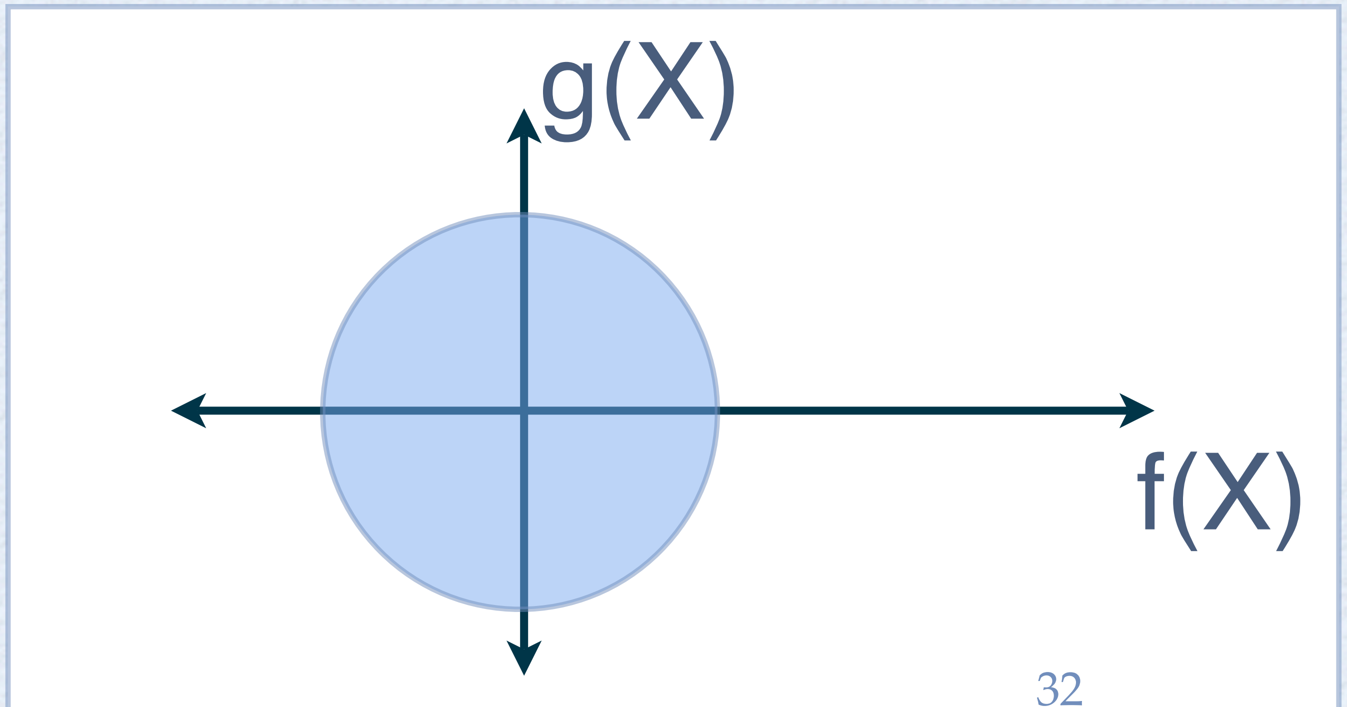
- allow comparing two continuous variables

```
(X, gaussian(0,1)) :: f(X).
(X, gaussian(5,3)) :: g(X).
h :- f(X), g(Y),above(X,Y).
```

Increase Expressivity (2)

- allow functions of multiple variables

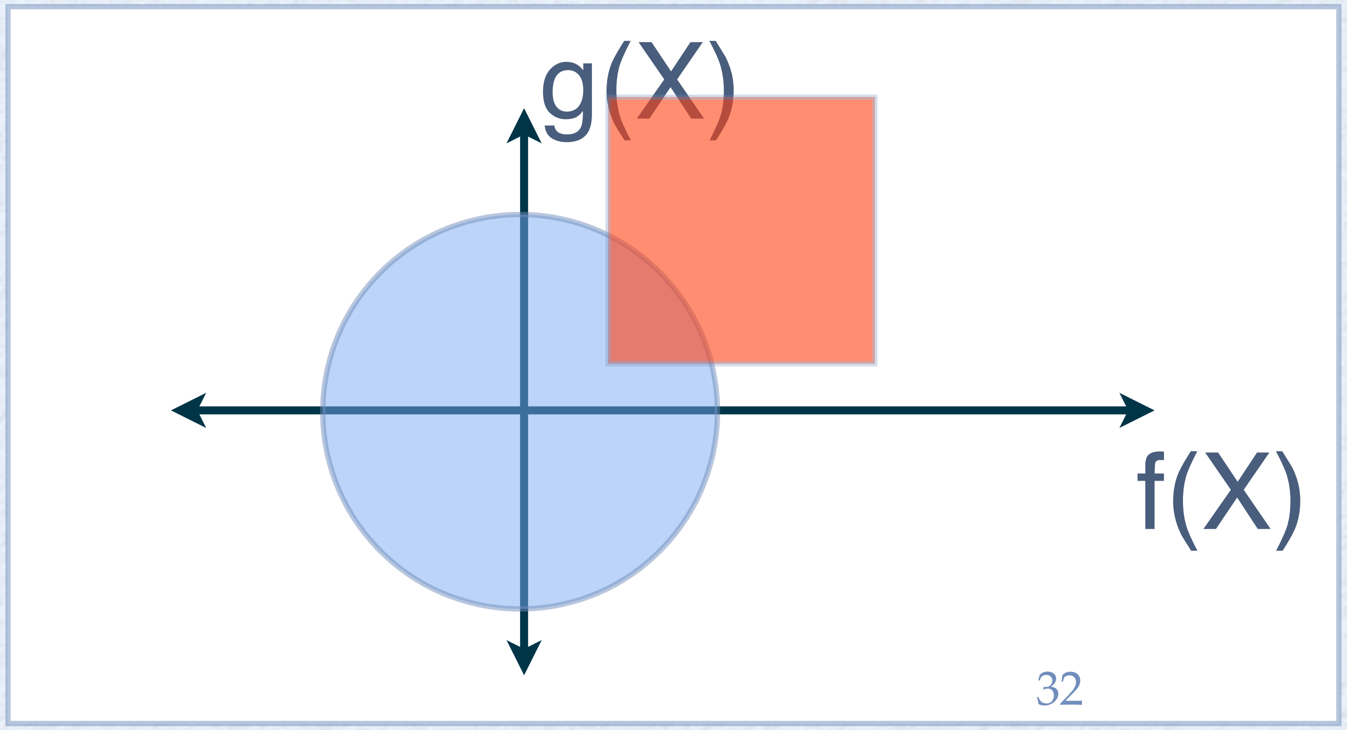
```
(X, gaussian(0,1)) :: f(X).
(X, gaussian(5,3)) :: g(X).
h :- f(X), g(Y), Z is X*Y,  below(Z,1)
```



Increase Expressivity (2)

- allow functions of multiple variables

```
(X, gaussian(0,1)) :: f(X).
(X, gaussian(5,3)) :: g(X).
h :- f(X), g(Y), Z is X*Y,  below(Z,1)
h :- f(X), g(Y), in_interval(X,0.5,2),
      in_interval(Y,0.5,2).
```



Conclusions

- Resulting language is more expressive than ProbLog
- quite restrictive to allow for exact inference
- suited for domains where proofs depend on a small number of continuous values
- Future Work
  - More expressivity
  - Improve learning

Thanks for your attention!
Questions?

ProbLog is available at
http://dtai.cs.kuleuven.be/problog/