CS 591: Formal Methods in Security and Privacy

Probabilistic Noninterference

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

From the previous classes

An example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
    cipher := msg xor key;
    return cipher
```

Learning a ciphertext does not change any a priori knowledge about the likelihood of messages.

Probabilistic While (PWhile)

```
c::= abort
   | skip
   | x:= e
   | x:=$ d
   | c;c
   | if e then c else c
   | while e do c
```

$d_1, d_2, \ldots$ probabilistic expressions

Semantics of Commands

This is defined on the structure of commands:

$\{\texttt{abort}\}_m = \mathbf{0}$

$\{\texttt{skip}\}_m = \texttt{unit}(m)$

$\{x := e\}_m = \texttt{unit}(m[x \leftarrow \{e\}_m])$

$\{x :=\$\ d\}_m = \texttt{let } a = \{d\}_m \texttt{ in unit}(m[x \leftarrow a])$

$\{c ; c'\}_m = \texttt{let } m' = \{c\}_m \texttt{ in } \{c'\}_{m'}$

$\{\texttt{if } e \texttt{ then } c_t \texttt{ else } c_f\}_m = \{c_t\}_m$ If $\{e\}_m = \texttt{true}$

$\{\texttt{if } e \texttt{ then } c_t \texttt{ else } c_f\}_m = \{c_f\}_m$ If $\{e\}_m = \texttt{false}$

$\{\texttt{while } e \texttt{ do } c\}_m = \texttt{sup}_{n \in \texttt{Nat}}\ \mu_n$

$\mu_n = \texttt{let } m' = \{(\texttt{while}^n\ e \texttt{ do } c)\}_m \texttt{ in } \{\texttt{if } e \texttt{ then abort}\}_{m'}$

Today:
Probabilistic
Noninterference

Revisiting the example

```
OneTimePad(m : private msg) : public msg
  key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

Learning a ciphertext does not change any a priori knowledge about the likelihood of messages.

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

Learning a ciphertext does not change any a priori knowledge about the likelihood of messages.

How do we formalize this?

Probabilistic Noninterference

A program `prog` is probabilistically noninterferent if and only if, whenever we run it on two low equivalent memories m_1 and m_2 we have that the probabilistic distributions we get as outputs are the same on public outputs.

Low equivalence on distributions

Two distributions over memories μ_1 and μ_2 are low equivalent if and only if they coincide after projecting out all the private variables.

In symbols: $\mu_1 \sim_{low} \mu_2$

Example: Low equivalence on distributions

Consider memories with x private and y public. The distributions μ_1 and μ_2 defined as:

μ_1`([x=2,y=0])=2/3,` μ_1`([x=3,y=1])=1/3`

and

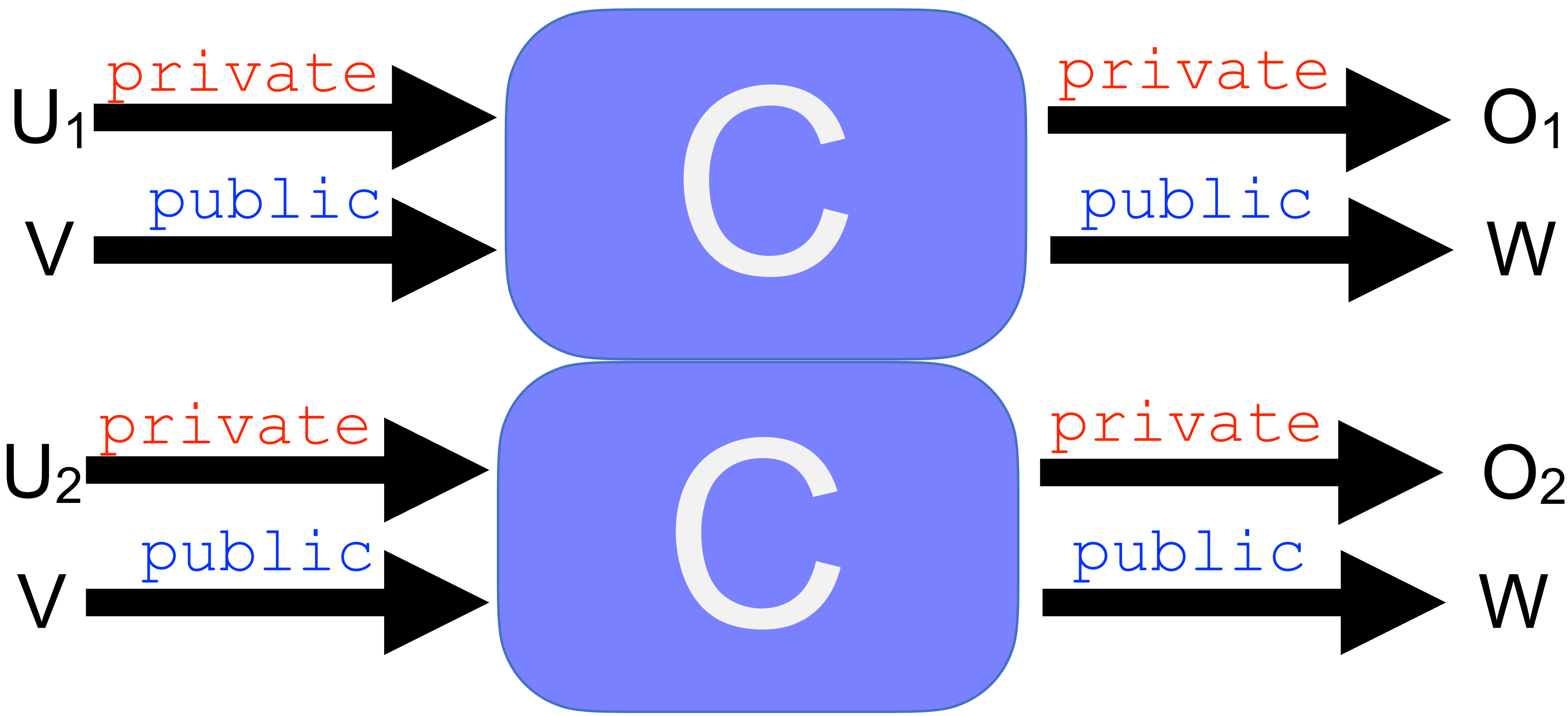
μ_2`([x=1,y=0])=1/3,`μ_2`([x=5,y=0])=1/3,`
μ_2`([x=4,y=1])=1/3`

are low equivalent.

Noninterference as a Relational Property

In symbols, c is noninterferent if and only if for every $m_1 \sim_{low} m_2$:

$\{c\}_{m1}=\mu_1$ and $\{c\}_{m2}=\mu_2$ implies $\mu_1 \sim_{low} \mu_2$



Revisiting the example

```
OneTimePad(m : private msg) : public msg
  key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

How can we prove that this is noninterferent?

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

m_1 m_2

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

m_1 m_2

↓

$m_1 \oplus k$

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

m_1 $\quad\quad\quad\quad$ m_2

↓

$m_1 \oplus k$

Suppose we can now chose the key for m_2. What could we choose?

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

m_1 → $m_1 \oplus k$

m_2 → $m_2 \oplus (m_1 \oplus k \oplus m_2)$

Suppose we can now chose the key for m_2. What could we choose?

Properties of xor

$$c \oplus (a \oplus c) = a$$

Properties of xor

$$c \oplus (a \oplus c) = a$$

Example:

$$100 \oplus (101 \oplus 100) =$$

$$100 \oplus 001 = 101$$

Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

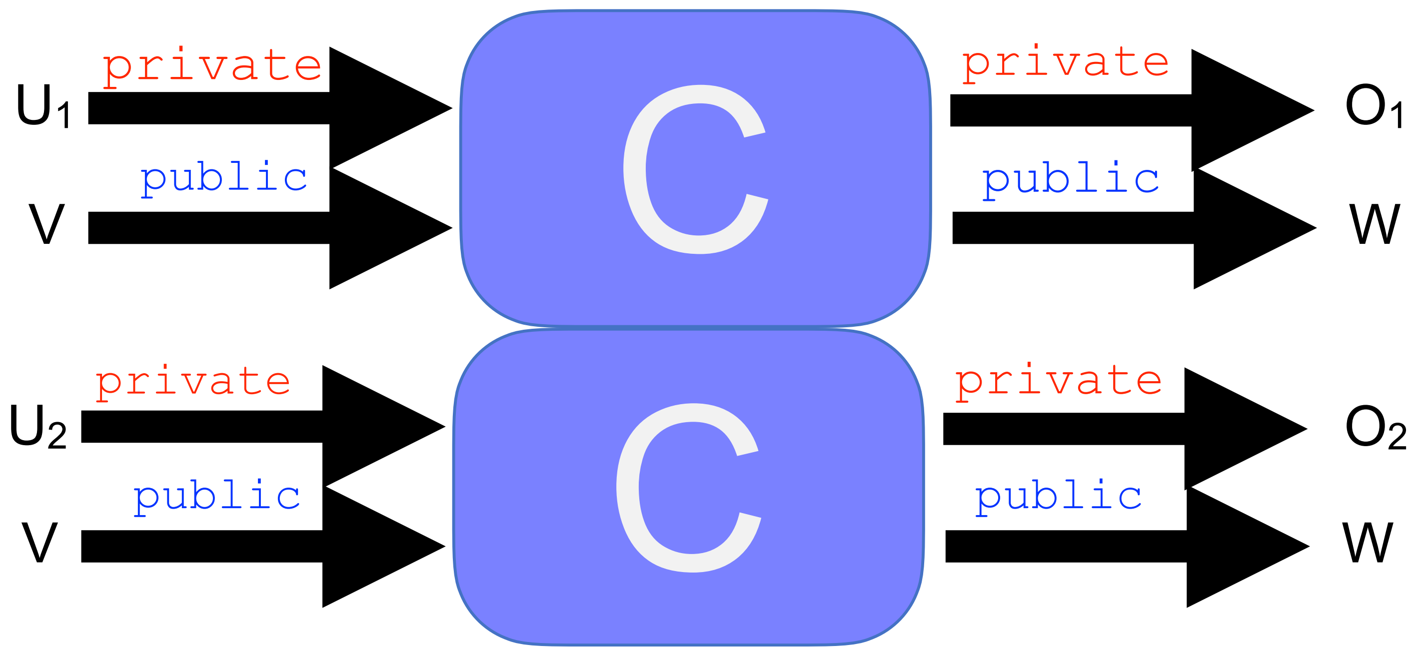
$m_1 \rightarrow m_1 \oplus k$

$m_2 \rightarrow m_1 \oplus k$

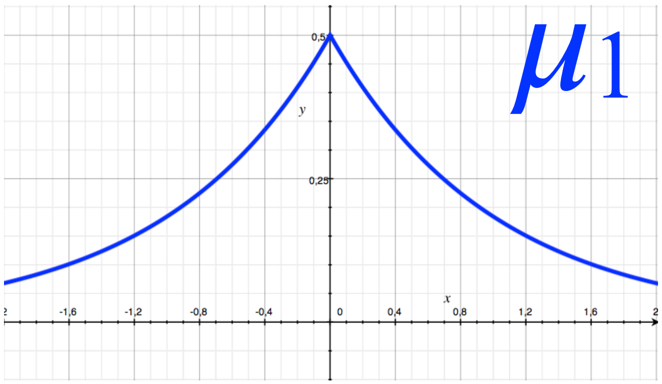
Applying the property above

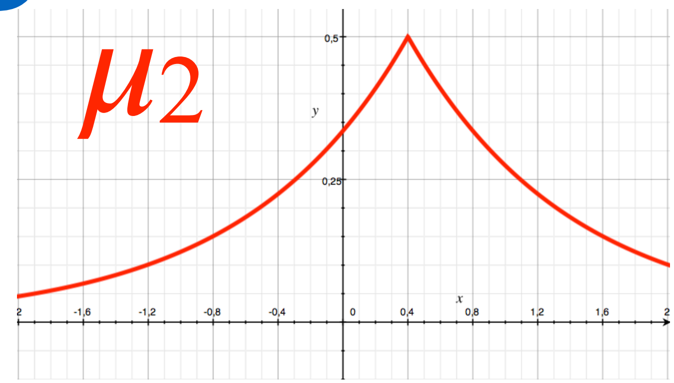
Revisiting the example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}ⁿ);
   cipher := msg xor key;
   return cipher
```

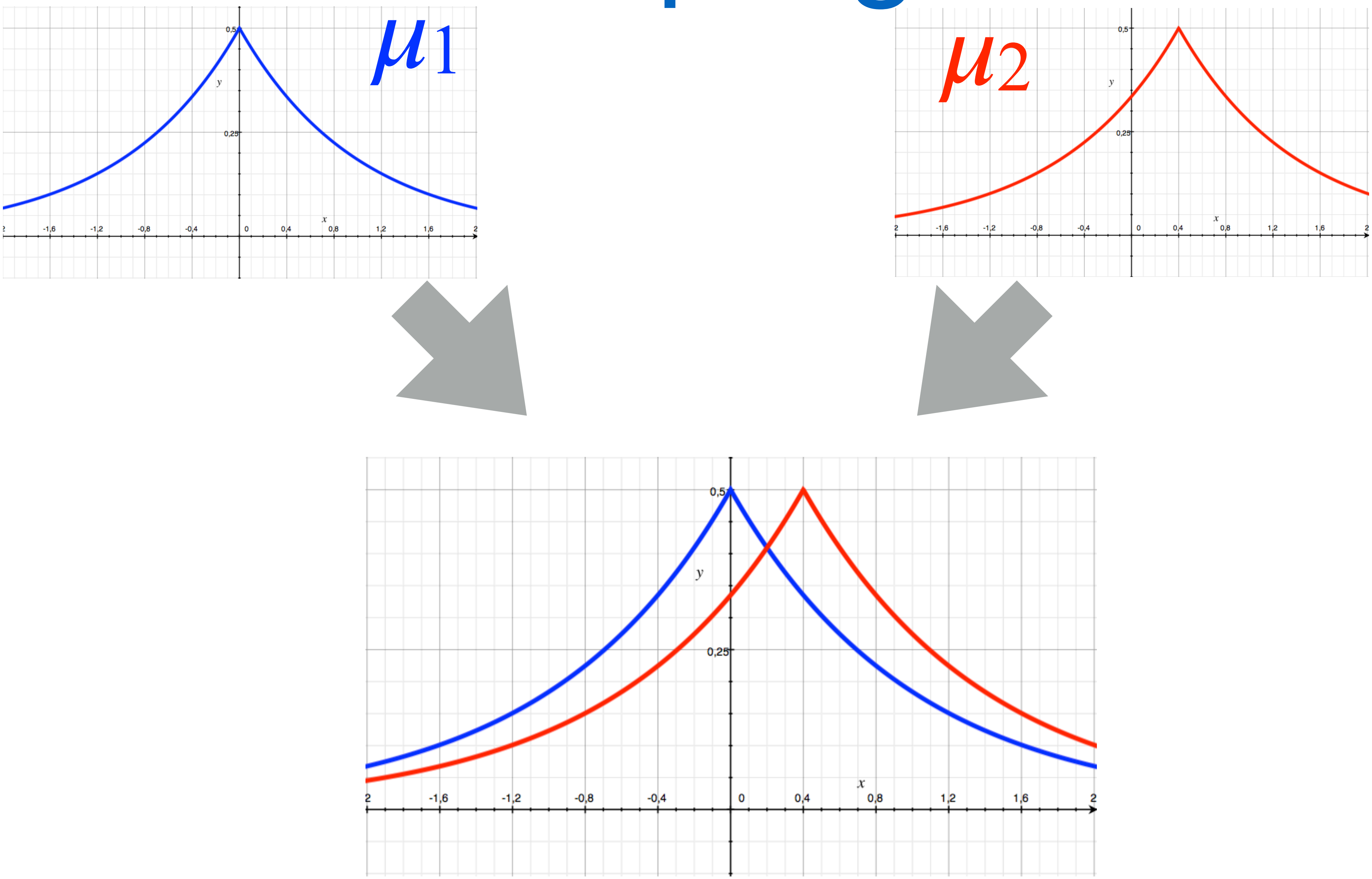


Coupling


μ1
0,5
0,25
y
x
-1,6
-1,2
-0,8
-0,4
0
0,4
0,8
1,2
1,6
2


μ2
0,5
0,25
y
x
-1,6
-1,2
-0,8
-0,4
0
0,4
0,8
1,2
1,6
2

Coupling



Example of Our Coupling

00	0.25
01	0.25
10	0.25
11	0.25

$$k_1 = 10 \oplus k_2 \oplus 00$$

00	0.25
01	0.25
10	0.25
11	0.25

Example of Our Coupling

OO	0.25
O1	0.25
1O	0.25
11	0.25

$$k_1 = 10 \oplus k_2 \oplus 00$$

OO	0.25
O1	0.25
1O	0.25
11	0.25

	OO	O1	1O	11
OO			0.25	
O1				0.25
1O	0.25			
11		0.25		

Coupling formally

Given two distributions $\mu_1 \in D(A)$, and $\mu_2 \in D(B)$, a coupling between them is a joint distribution $\mu \in D(A \times B)$ whose marginal distributions are μ_1 and μ_2, respectively.

$$\pi_1(\mu)(a) = \sum_b \mu(a, b) \qquad \pi_2(\mu)(b) = \sum_a \mu(a, b)$$

Probabilistic Relational Hoare Quadruples

Precondition

Program$_1$ ~ Program$_2$

Postcondition

Precondition

$$c_1 \sim c_2 : P \Rightarrow Q$$

Probabilistic Program

Probabilistic Program

Postcondition

Validity of Probabilistic Hoare quadruple

We say that the quadruple $c_1 \sim c_2 : P \Rightarrow Q$ is valid if and only if for every pair of memories m_1, m_2 such that $P(m_1, m_2)$ we have: $\{c_1\}_{m1} = \mu_1$ and $\{c_2\}_{m2} = \mu_2$ implies $Q(\mu_1, \mu_2)$.

Validity of Probabilistic Hoare quadruple

We say that the quadruple $c_1 \sim c_2 : P \Rightarrow Q$ is valid if and only if for every pair of memories m_1, m_2 such that $P(m_1, m_2)$ we have:

$\{c_1\}_{m1} = \mu_1$ and $\{c_2\}_{m2} = \mu_2$ implies $Q(\mu_1, \mu_2)$.

Is this correct?!?

Relational Assertions

$$c_1 \sim c_2 : P \Rightarrow Q$$

P: logical formula over pair of memories (i.e. relation over memories)

Q: logical formula over ????

R-Coupling

Given two distributions $\mu_1 \in D(A)$, and $\mu_2 \in D(B)$, an R-coupling between them, for $R \subseteq A \times B$, is a joint distribution $\mu \in D(A \times B)$ such that:

1) the marginal distributions of μ are μ_1 and μ_2, respectively,
2) the support of μ is contained in R. That is, if μ`(a,b)>0, then (a,b)`$\in$R.

Relational lifting of a predicate

We say that two subdistributions $\mu_1 \subseteq D(A)$ and $\mu_2 \subseteq D(B)$ are in the relational lifting of the relation $R \subseteq A x B$, denoted μ_1 R* μ_2 if and only if there exist a subdistribution $\mu \subseteq D(AxB)$ such that:

1) if μ`(a,b)>0, then (a,b)`$\in$`Q`.

2) $\pi_1(\mu) = \mu_1$ and $\pi_2(\mu) = \mu_2$

Relational lifting of a predicate

We say that two subdistributions $\mu_1 \subseteq D(A)$ and $\mu_2 \subseteq D(B)$ are in the relational lifting of the relation $R \subseteq A x B$, denoted μ_1 R* μ_2 if and only if there exist a subdistribution $\mu \subseteq D(AxB)$ such that:

1) if $\mu(a,b)>0$, then $(a,b) \in Q$.

2) $\pi_1(\mu) = \mu_1$ and $\pi_2(\mu) = \mu_2$

Does it remind you something?

Validity of Probabilistic Hoare quadruple

We say that the quadruple $c_1 \sim c_2 : P \Rightarrow Q$ is valid if and only if for every pair of memories m_1, m_2 such that $P(m_1, m_2)$ we have:

$\{c_1\}_{m1} = \mu_1$ and $\{c_2\}_{m2} = \mu_2$ implies $Q^*(\mu_1, \mu_2)$.

Probabilistic Relational Hoare Logic

Skip

$$\frac{}{\vdash \texttt{skip} \sim \texttt{skip} : P \Rightarrow P}$$

Probabilistic Relational Hoare Logic

Assignment

$$\vdash x_1 := e_1 \sim x_2 := e_2 : P[e_1\langle 1\rangle / x_1\langle 1\rangle, e_2\langle 2\rangle / x_2\langle 2\rangle] \Rightarrow P$$

Probabilistic Relational Hoare Logic

Composition

$$\frac{\vdash c_1 \sim c_2 : P \Rightarrow R \qquad \vdash c_1' \sim c_2' : R \Rightarrow S}{\vdash c_1; c_1' \sim c_2; c_2' : P \Rightarrow S}$$

Probabilistic Relational Hoare Logic

Consequence

$$\frac{P \Rightarrow S \qquad \vdash c_1 \sim c_2 : S \Rightarrow R \qquad R \Rightarrow Q}{\vdash c_1 \sim c_2 : P \Rightarrow Q}$$

We can weaken P, i.e. replace it by something that is implied by P. In this case S.

We can strengthen Q, i.e. replace it by something that implies Q. In this case R.

Probabilistic Relational Hoare Logic

If-then-else

$$P \Rightarrow (e_1\langle 1\rangle \Leftrightarrow e_2\langle 2\rangle)$$

$$\vdash c_1 \sim c_2 : e_1\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash c_1' \sim c_2' : \neg e_1\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash \begin{array}{c} \texttt{if } e_1 \texttt{ then } c_1 \texttt{ else } c_1' \\ \sim \\ \texttt{if } e_2 \texttt{ then } c_2 \texttt{ else } c_2' \end{array} : P \Rightarrow Q$$

Probabilistic Relational Hoare Logic

While

$$P \Rightarrow (e_1\langle 1\rangle \Leftrightarrow e_2\langle 2\rangle)$$

$$\frac{\vdash c_1 \sim c_2 \;:\; e_1\langle 1\rangle \wedge P \Rightarrow P}{\vdash \begin{array}{c}\texttt{while } e_1 \texttt{ do } c_1 \\ \sim \\ \texttt{while } e_2 \texttt{ do } c_2\end{array} : P \Rightarrow P \wedge \neg e_1\langle 1\rangle}$$

Probabilistic Relational Hoare Logic

If-then-else - left

$$\frac{\vdash c_1 \sim c_2 : e\langle 1\rangle \wedge P \Rightarrow Q \qquad \vdash c_1' \sim c_2 : \neg e\langle 1\rangle \wedge P \Rightarrow Q}{\vdash \begin{array}{c}\texttt{if } e \texttt{ then } c_1 \texttt{ else } c_1' \\ \sim \\ c_2\end{array} : P \Rightarrow Q}$$

Probabilistic Relational Hoare Logic

If-then-else - right

$$\vdash c_1 \sim c_2 : e\langle 2\rangle \wedge P \Rightarrow Q$$

$$\vdash c_1 \sim c_2' : \neg e\langle 2\rangle \wedge P \Rightarrow Q$$

$$\vdash \begin{array}{c} c_1 \\ \sim \\ \texttt{if } e \texttt{ then } c_2 \texttt{ else } c_2' \end{array} : P \Rightarrow Q$$

Probabilistic Relational Hoare Logic

Assignment - left

```
⊢x:=e ~ skip:
  P[e<1>/x<1>] ⇒ P
```

How about the random assignment?

Probabilistic Relational Hoare Logic

Random Assignment

$$\vdash \texttt{x}_1 \texttt{ :=\$ d}_1 \sim \texttt{x}_2 \texttt{ :=\$ d}_2 : \texttt{??}$$

We would like to have:

$$\frac{\begin{array}{l}\texttt{P}(m_1,m_2)\\ \Rightarrow\\ \texttt{let a=\{}d_1\texttt{\}}_{m1}\ \texttt{in unit}(m_1[x_1\leftarrow a])\\ Q^*\\ \texttt{let a=\{}d_2\texttt{\}}_{m2}\ \texttt{in unit}(m_2[x_2\leftarrow a])\end{array}}{\vdash x_1 \texttt{ :=\$ } d_1 \sim x_2 \texttt{ :=\$ } d_2 : P \Rightarrow Q}$$

What is the problem with this rule?

Restricted Probabilistic Expressions

We consider a restricted set of expressions denoting probability distributions.

```
d::= f(d1,…,dk)
```

Where `f` is a distribution declaration

Some expression examples similar to the previous

`uniform({0,1}^128)` `bernoulli(.5)` `laplace(0,1)`

Restricted Probabilistic Expressions

We consider a restricted set of expressions denoting probability distributions.

```
d::= f(d1,…,dk)
```

Where `f` is a distribution declaration

Some expression examples similar to the previous

`uniform({0,1}^128)`   `bernoulli(.5)`   `laplace(0,1)`

Notice that we don't need a memory anymore to interpret them

A sufficient condition for R-Coupling

Given two distributions $\mu_1 \in D(A)$, and $\mu_2 \in D(B)$, and a relation $R \subseteq A\text{x}B$, if there is a mapping $h:A \rightarrow B$ such that:

1) h is a bijective map between elements in $supp(\mu_1)$ and $supp(\mu_2)$,
2) for every $a \in supp(\mu_1)$, $(a,h(a)) \in R$
3) $Pr_{x\sim\mu 1}[x=a]=Pr_{x\sim\mu 2}[x=h(a)]$

Then, there is an R-coupling between μ_1 and μ_2.
We write $h \triangleleft (\mu_1,\mu_2)$ in this case.

Probabilistic Relational Hoare Logic
Random Assignment

$$\frac{h \lhd (\{d_1\}, \{d_2\}) \quad P = \forall v, v \in supp(\{d_1\}) \Rightarrow Q[v/x_1\langle 1\rangle, h(v)/x_2\langle 2\rangle]}{\vdash x_1 := \$\ d_1 \sim x_2 := \$\ d_2 : P \Rightarrow Q}$$

Back to our example

```
OneTimePad(m : private msg) : public msg
  key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

Back to our example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

Back to our example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

m_1 m_2

Back to our example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

m_1 m_2

↓

$m_1 \oplus k$

Back to our example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

$m_1 \rightarrow m_1 \oplus k$

$m_2 \rightarrow m_2 \oplus (m_1 \oplus k \oplus m_2)$

Back to our example

```
OneTimePad(m : private msg) : public msg
    key :=$ Uniform({0,1}^n);
    cipher := msg xor key;
    return cipher
```

$d_1 = \texttt{Uniform}(\{0,1\}^n)$ $d_2 = \texttt{Uniform}(\{0,1\}^n)$

Is this a good map?

$$h(k) = (m\langle 1\rangle \oplus k \oplus m\langle 2\rangle)$$

Back to our example

```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

d_1=Uniform({0,1}^{n})

d_2=Uniform({0,1}^{n})

Is this a good map?

$$h(k) = (m\langle 1\rangle \oplus k \oplus m\langle 2\rangle)$$

What is the relation?

Back to our example

```
OneTimePad(m : private msg) : public msg
  key :=$ Uniform({0,1}ⁿ);
  cipher := msg xor key;
  return cipher
```

d_1=Uniform({0,1}^{n})

d_2=Uniform({0,1}^{n})

Is this a good map?

$$h(k) = (m\langle 1\rangle \oplus k \oplus m\langle 2\rangle)$$

What is the relation?

$$m\langle 1\rangle \oplus k\langle 1\rangle = m\langle 2\rangle \oplus k\langle 2\rangle$$

Back to our example

d_1=Uniform($\{0,1\}^n$)

d_2=Uniform($\{0,1\}^n$)

Is this a good map?

$$h(k) = (m<1> \oplus k \oplus m<2>)$$

1) it is bijective between elements in the support of $\{d_1\}$ and $\{d_2\}$
2) for every $k \in supp(\{d_1\})$, $m<1> \oplus k = m<2> \oplus (m<1> \oplus k \oplus m<2>)$
3) $Pr_{x\sim\{d1\}}[x=v] = Pr_{x\sim\{d2\}}[x=v]$

Back to our example

d_1=Uniform($\{0,1\}^n$) d_2=Uniform($\{0,1\}^n$)

Is this a good map?

$$h(k) = (m\text{<}1\text{>} \oplus k \oplus m\text{<}2\text{>})$$

1) it is bijective between elements in the support of $\{d_1\}$ and $\{d_2\}$
2) for every $k \in$ supp($\{d_1\}$), $m\text{<}1\text{>} \oplus k = m\text{<}2\text{>} \oplus (m\text{<}1\text{>} \oplus k \oplus m\text{<}2\text{>})$
3) $Pr_{x\sim\{d1\}}[x=v] = Pr_{x\sim\{d2\}}[x=v]$

It is a good map!

Back to our example

$$h(k) = (m\langle 1\rangle \oplus k \oplus m\langle 2\rangle) \lhd (\{d_1\},\{d_2\})$$

$$P = \forall k, k \in \{0,1\}^n$$

$$\Rightarrow\ m\langle 1\rangle \oplus k_1\langle 1\rangle = m\langle 2\rangle \oplus k_2\langle 2\rangle [v/k_1\langle 1\rangle, h(v)/k_2\langle 2\rangle] = m\langle 1\rangle \oplus k = m\langle 2\rangle \oplus (m\langle 1\rangle \oplus k \oplus m\langle 2\rangle)$$

$$\vdash k_1 := \$Uniform(\{0,1\}^n) \sim k_2 := \$Uniform(\{0,1\}^n) : True \Rightarrow m\langle 1\rangle \oplus k_1\langle 1\rangle = m\langle 2\rangle \oplus k_2\langle 2\rangle$$

Back to our example

$$h(k) = (m<1> \oplus k \oplus m<2>) \lhd (\{d_1\},\{d_2\})$$

$$P = \forall k, k \in \{0,1\}^n$$

$$\Rightarrow\ m<1> \oplus k_1<1> = m<2> \oplus k_2<2>[v/k_1<1>, h(v)/k_2<2>] = m<1> \oplus k = m<2> \oplus (m<1> \oplus k \oplus m<2>)$$

$$\vdash k_1 := \$Uniform(\{0,1\}^n) \sim k_2 := \$Uniform(\{0,1\}^n) : True\ \Rightarrow\ m<1> \oplus k_1<1> = m<2> \oplus k_2<2>$$

Using the assignment rule, we can conclude.

Soundness

If we can derive $\vdash c_1 \sim c_2 : P \Rightarrow Q$ through the rules of the logic, then the quadruple

$c_1 \sim c_2 : P \Rightarrow Q$ is valid.

Completeness?