CS 591: Formal Methods in Security and Privacy

Probabilistic relational Hoare Logic

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

CS 591: Formal Methods in Security and Privacy

Probabilistic relational Hoare Logic

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

Zoom
Participants Cameras

Projects

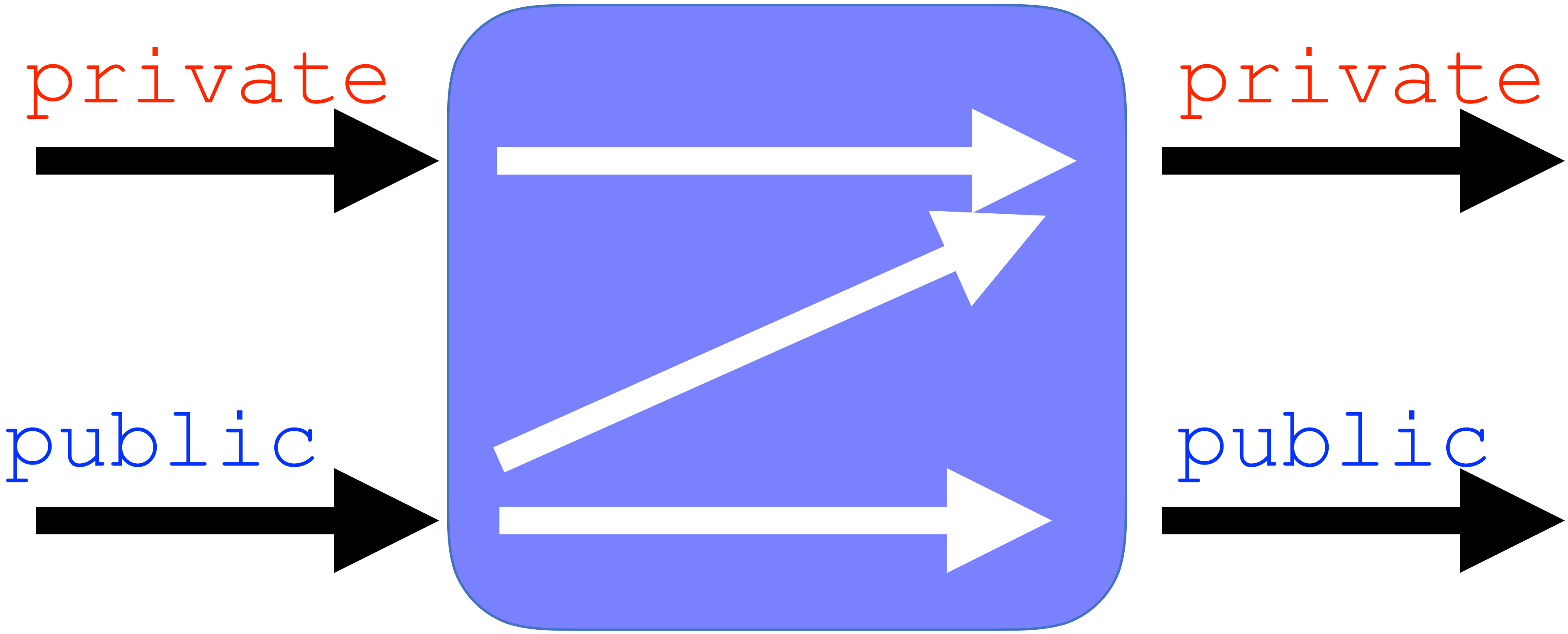
By the end of the week, everyone should know what to work on for the project.

If you don’t know yet what you want to work on, let’s schedule a time by email to zoom with Alley and me about projects ideas.

From the previous classes

Information Flow Control

We want to guarantee that confidential inputs do not flow to nonconfidential outputs.



Does this program satisfy noninterference?

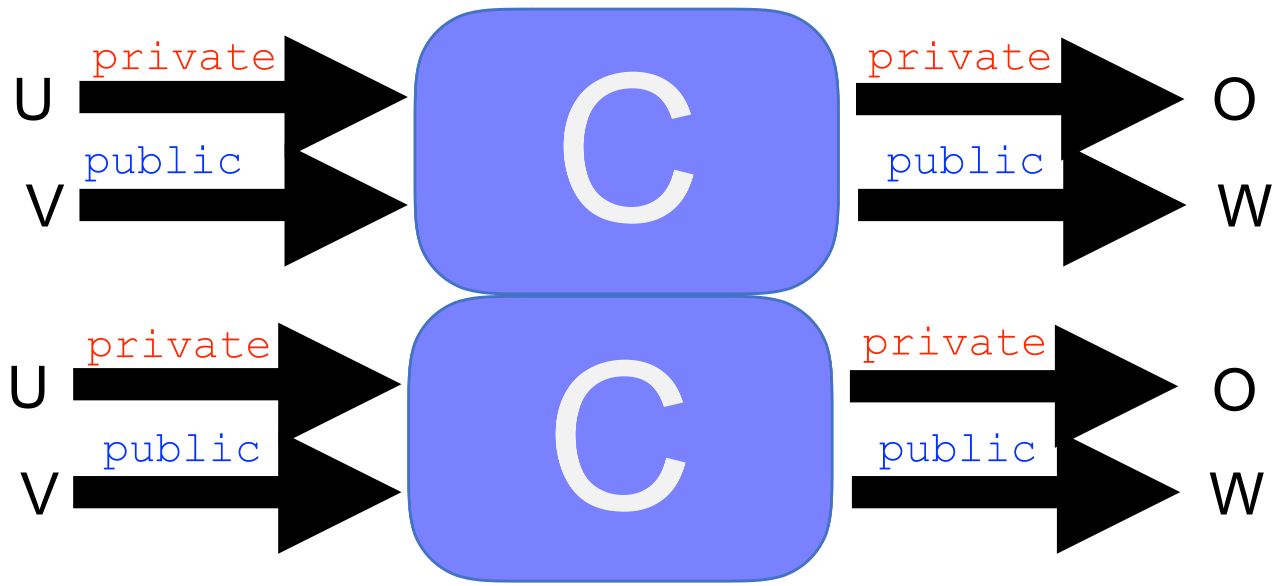
```
s1:public
s2:private
r:private
i:public

proc Compare (s1:list[n] bool,s2:list[n] bool)
i:=0;
r:=0;
while i<n do
 if not(s1[i]=s2[i]) then
     r:=1
 i:=i+1
```

Noninterference as a Relational Property

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

1) $\{c\}_{m1} = \perp$ iff $\{c\}_{m2} = \perp$
2) $\{c\}_{m1} = m_1'$ and $\{c\}_{m2} = m_2'$ implies $m_1' \sim_{low} m_2'$



Relational Hoare Quadruples

Precondition

(a logical formula)

Precondition
`Program1 ~ Program2`
Postcondition

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

Program Program Postcondition

(a logical formula)

Soundness

If we can derive $\vdash \texttt{c}_1 \sim \texttt{c}_2 : P \Rightarrow Q$ through the rules of the logic, then the quadruple $\texttt{c}_1 \sim \texttt{c}_2 : P \Rightarrow Q$ is valid.

Relative Completeness

If a quadruple $\mathtt{c}_1 \sim \mathtt{c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}$ is valid, and we have an oracle to derive all the true statements of the form $\mathtt{P} \Rightarrow \mathtt{S}$ and of the form $\mathtt{R} \Rightarrow \mathtt{Q}$, then we can derive $\vdash \mathtt{c}_1 \sim \mathtt{c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}$ through the rules of the logic.

Soundness and completeness with respect to Hoare Logic

$$\vdash_{\mathtt{RHL}}\ \mathtt{c}_1 \sim \mathtt{c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}$$

iff

$$\vdash_{\mathtt{HL}}\ \mathtt{c}_1 ; \mathtt{c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}$$

Under the assumption that we can partition the memory adequately, and that we have termination.

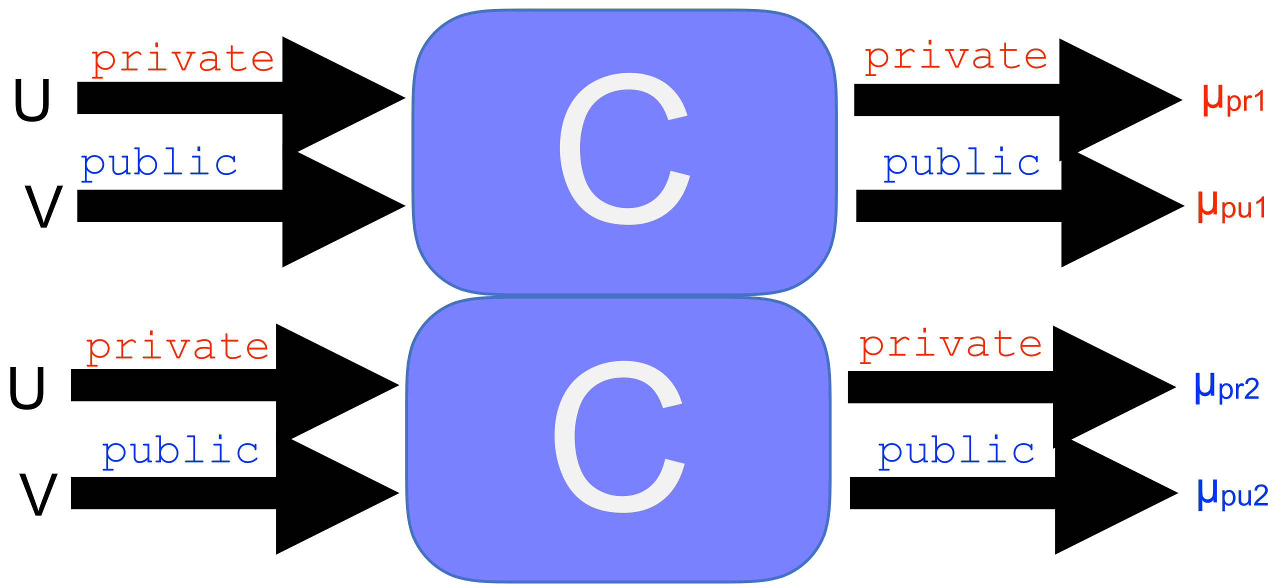
Probabilistic Noninterference

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.

Probabilistic Noninterference as a Relational Property

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

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



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.

Semantics of Commands

This is defined on the structure of commands:

$\{\texttt{abort}\}_m = \mathbf{0}$ $\qquad\{\texttt{skip}\}_m = \texttt{unit(m)}$

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

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

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

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

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

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

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

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 $\quad$ m_2

$$m_1 \downarrow$$

$m_1 \oplus k$

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
```

$m_1 \longrightarrow m_1 \oplus k$

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

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
```

Applying the property above

Revisiting the example

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

m_1 m_2

Applying the property above

Revisiting the example

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

m_1 $\longrightarrow$ $m_1 \oplus k$

m_2

Applying the property above

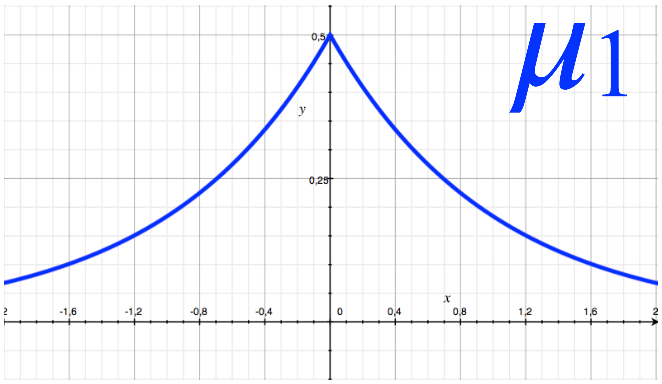
Revisiting the example

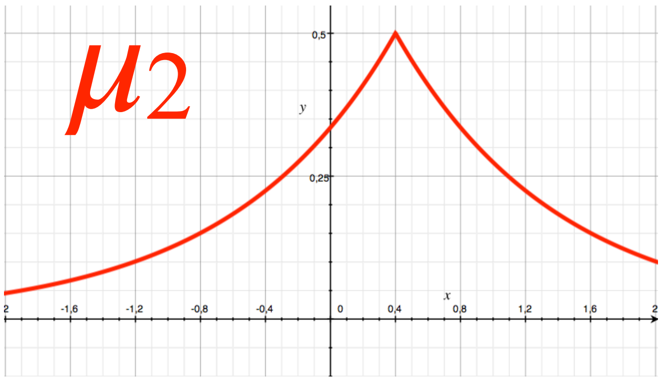
```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}ⁿ);
   cipher := msg xor key;
   return cipher
```

$$\begin{array}{ccc} m_1 & & m_2 \\ \downarrow & & \downarrow \\ m_1 \oplus k & & m_1 \oplus k \end{array}$$

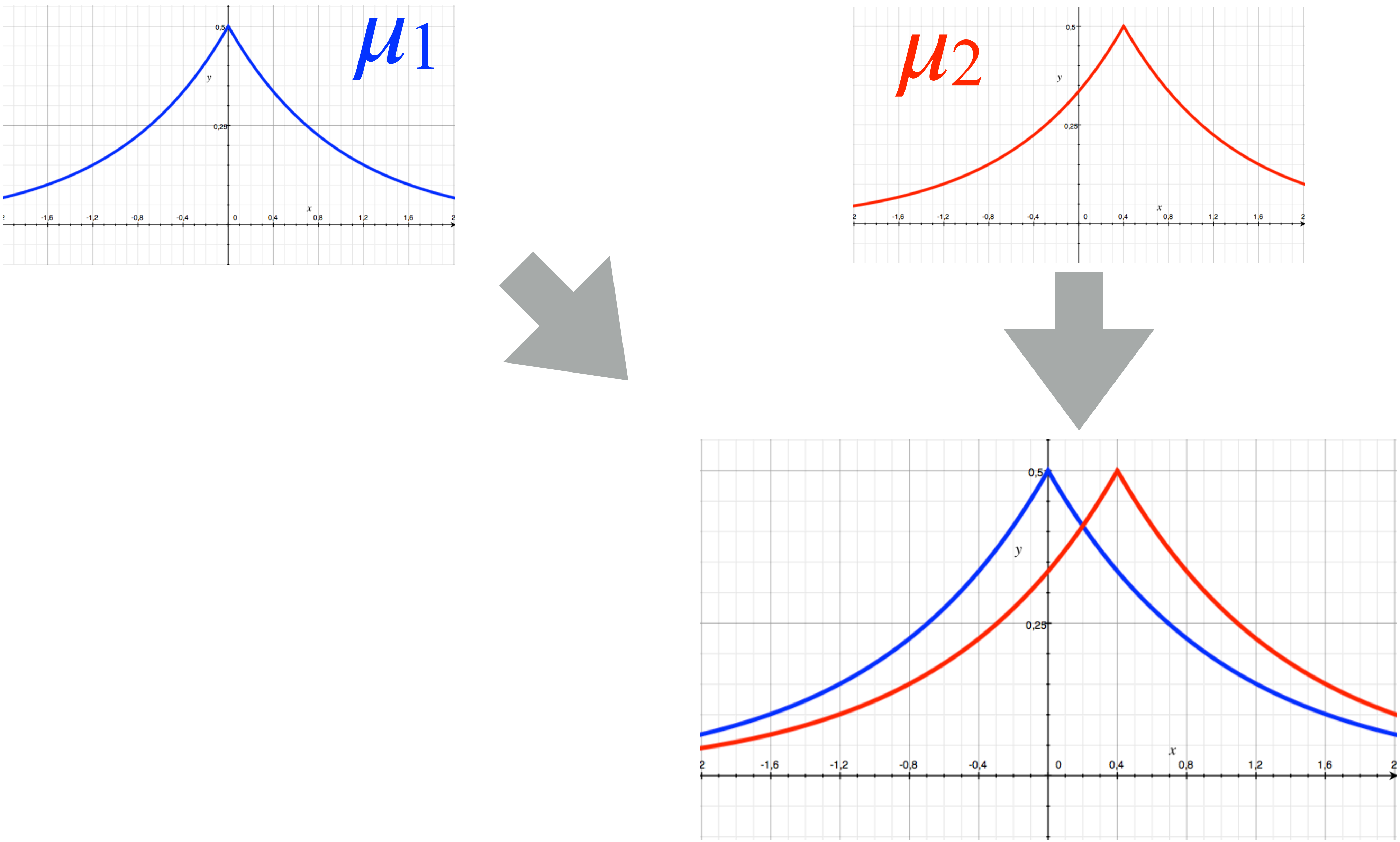
Applying the property above

Coupling





Coupling



Example of Our Coupling

00	0.25
01	0.25
10	0.25
11	0.25

$$k = 10 \oplus k \oplus 00$$

00	0.25
01	0.25
10	0.25
11	0.25

Example of Our Coupling

00	0.25
01	0.25
10	0.25
11	0.25

$$k = 10 \oplus k \oplus 00$$

00	0.25
01	0.25
10	0.25
11	0.25

	00	01	10	11
00			0.25	
01				0.25
10	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.

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

Today:
Probabilistic Relational Hoare Logic

Probabilistic Relational Hoare Quadruples

Precondition
Program$_1$ ~ Program$_2$
Postcondition

Precondition
(a logical formula)

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

Probabilistic Program

Probabilistic Program

Postcondition
(a logical formula)

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

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

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

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 $\mu(a,b) > 0$, 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 $\mu_1 \ R^* \ \mu_2$ 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 $\pi_2(\mu) = \mu_2$

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 $R \subseteq AxB$, 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

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

Probabilistic Relational Hoare Logic

Composition

$$\dfrac{\vdash \texttt{c}_1 \sim \texttt{c}_2 : \texttt{P} \Rightarrow \texttt{R} \qquad \vdash \texttt{c}_1' \sim \texttt{c}_2' : \texttt{R} \Rightarrow \texttt{S}}{\vdash \texttt{c}_1 ; \texttt{c}_1' \sim \texttt{c}_2 ; \texttt{c}_2' : \texttt{P} \Rightarrow \texttt{S}}$$

Probabilistic Relational Hoare Logic

Consequence

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

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

$$\frac{P \Rightarrow (e_1\langle 1\rangle \Leftrightarrow e_2\langle 2\rangle) \qquad \vdash c_1 \sim c_2 : e_1\langle 1\rangle \wedge P \Rightarrow P}{\vdash \begin{array}{c}\texttt{while } e_1 \texttt{ do } c_1 \\ \sim \\ \texttt{while } e_2 \texttt{ do } c_2\end{array} : P \Rightarrow P \wedge \neg e_1\langle 1\rangle}$$

Probabilistic Relational Hoare Logic

If-then-else - left

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

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

$$\dfrac{}{\vdash \texttt{x:=e} \sim \texttt{skip:}\ \texttt{P[e<1>/x<1>]} \Rightarrow \texttt{P}}$$

How about the random assignment?

Probabilistic Relational Hoare Logic

Random Assignment

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

We would like to have:

$$\frac{\begin{array}{l}\texttt{P}(\texttt{m}_1,\texttt{m}_2) \\ \Rightarrow \\ \texttt{let a=}\{\texttt{d}_1\}_{\texttt{m1}}\ \texttt{in unit}(\texttt{m}_1[\texttt{x}_1\leftarrow\texttt{a}]) \\ \texttt{Q*} \\ \texttt{let a=}\{\texttt{d}_2\}_{\texttt{m2}}\ \texttt{in unit}(\texttt{m}_2[\texttt{x}_2\leftarrow\texttt{a}])\end{array}}{\vdash \texttt{x}_1 \texttt{ :=\$ } \texttt{d}_1 \sim \texttt{x}_2 \texttt{ :=\$ } \texttt{d}_2 : \texttt{P} \Rightarrow \texttt{Q}}$$

We would like to have:

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

What is the problem with this rule?