CS 511: Formal Methods in Security and Privacy

Accuracy in Differential Privacy

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

(ε,δ)-Differential Privacy

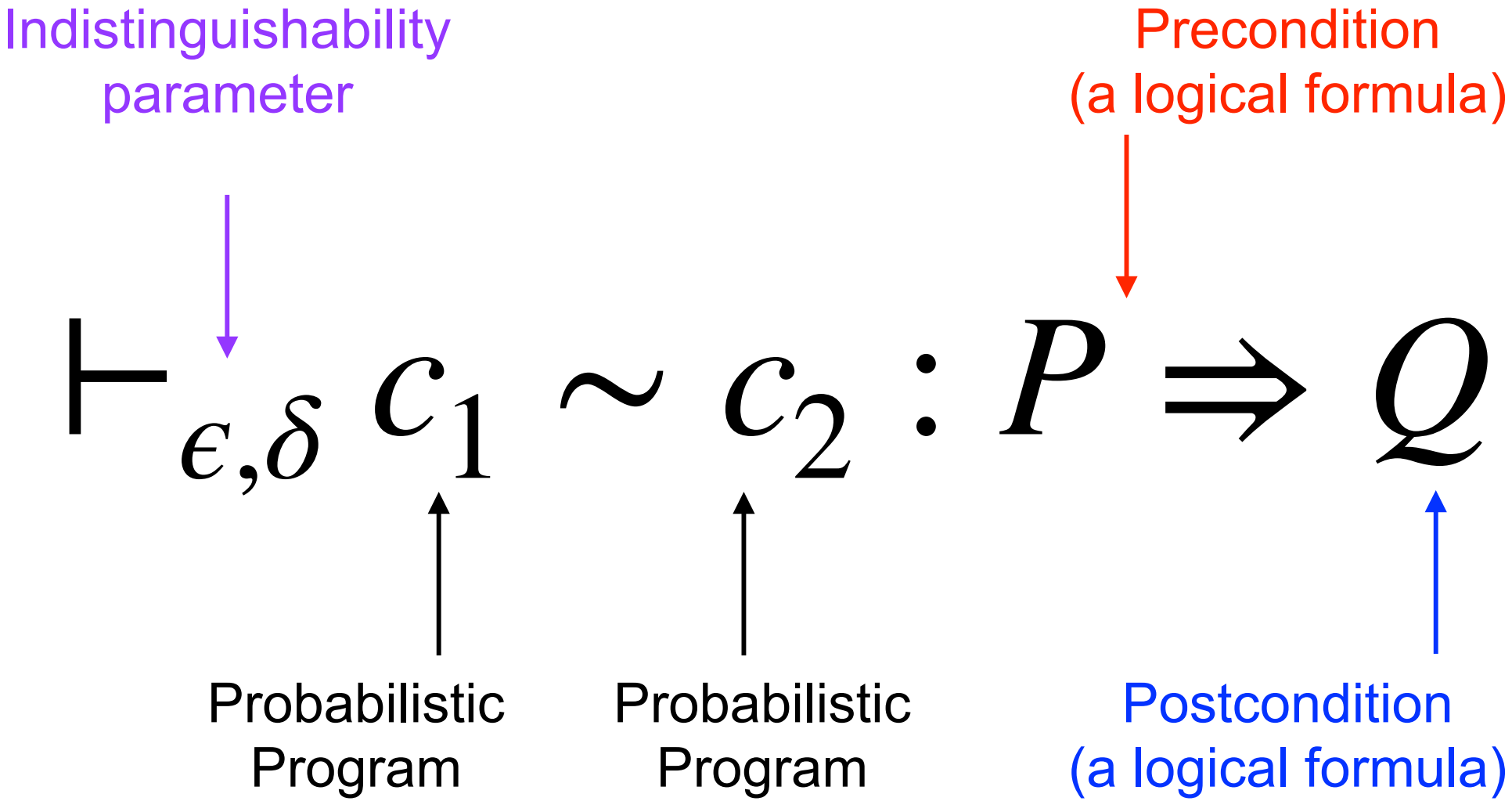
Definition

Given $\varepsilon, \delta \geq 0$, a probabilistic query $Q: X^n \rightarrow R$ is (ε,δ)-differentially private iff

for all adjacent database b_1, b_2 and for every $S \subseteq R$:

$$\Pr[Q(b_1) \in S] \leq \exp(\varepsilon)\Pr[Q(b_2) \in S] + \delta$$

apRHL


Indistinguishability parameter
Precondition (a logical formula)
$\vdash_{\epsilon,\delta} c_1 \sim c_2 : P \Rightarrow Q$
Probabilistic Program
Probabilistic Program
Postcondition (a logical formula)

Laplace Mechanism

```
Lap(d : priv data)(f: data -> real)
    (e:real) : pub real
   z:=$ Laplace(GS_f/e,0)
   z:= f(d)+z
   return z
```

Releasing privately the mean of Some Data

```
Mean(d : {0,1} list) : real
   i:=0;
   s:=0;
   n:=size(d);
   while (i<n)
      s:=s + d[i]
      i:=i+1;
   z:=$ Laplace(1/(n*eps),0)
   z:= (s/n)+z
   return z
```

Releasing partial sums - v1

```
DummySum(d : {0,1} list) : real list
   i:= 0;
   s:= 0;
   r:= [];
   while (i<size d)
      s:= s + d[i]
      z:=$ Lap(eps,s)
      r:= r ++ [z];
      i:= i+1;
   return r
```

Releasing partial sums - v2

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

Report Noisy Max

```
RNM (q_1,...,q_N : (data → R) list,
     b : list data, ε: R) : nat
  i = 0;
  max = 0;
  while (i < N){
     cur = q_i(b) + Lap(1/ε)
     if (cur > max)
          max =  cur ;
          output = i;
  return output;
```

How can we reason about the accuracy of these mechanisms?

How about accuracy?

We have several ways to estimate how good is a noised query answer. Two examples:

- by comparing the noised answer with the output of the noise-free computation
- by comparing the noised answer with the expected value of the analysis on the population

Estimating the Error

$$\Pr_{r \sim M(D)}[\, u(q(D), r) \geq \alpha] \leq \beta$$

Estimating the Error

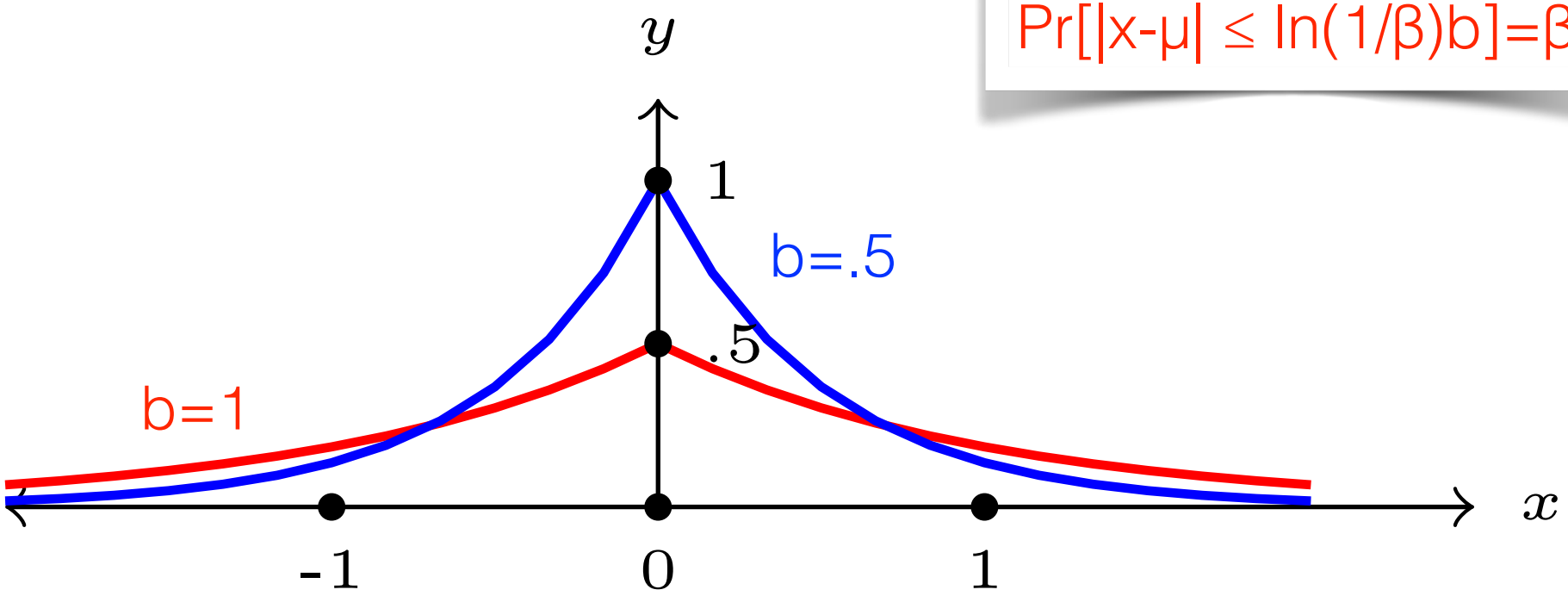
$$\Pr_{r \sim M(D)}[\, u(q(D), r) \geq \alpha] \leq \beta$$

We could be interested in showing that with probability **1-β** a mechanism returns a value that is α-close to the original one.

Laplace' tail bound

$$\mathsf{Lap}(b,\mu)(X) = \frac{1}{2b}\exp\left(-\frac{|\mu - X|}{b}\right)$$

$\Pr[|x-\mu| \leq \ln(1/\beta)b] = \beta$



Laplace Mechanism utility

```
Lap(d : priv data)(f: data -> real)
    (e:real) : pub real
   z:=$ Laplace(GS_f/e,0)
   z:= f(d)+z
   return z
```

$$\Pr_{r \sim Lap(x)}[\,|f(x) - r| \geq \ln(\frac{1}{\beta})\frac{GS_f}{\epsilon}] \leq \beta$$

Mean Utility

```
Mean d : {0,1} list) : real
   i:=0;
   s:=0;
   n:=size(d);
   while (i<n)
       s:=s + d[i]
       i:=i+1;
   z:=$ Laplace(1/(n*eps),0)
   z:= (s/n)+z
   return z
```

$$\Pr_{r\sim Lap(x)}[\,|f(x)-r|\geq \ln(\frac{1}{\beta})\frac{1}{n\epsilon}]\leq\beta$$

DummySum utility

```
DummySum(d : {0,1} list) : real list
   i:= 0;
   s:= 0;
   r:= [];
   while (i<size d)
      s:= s + d[i]
      z:=$ Lap(eps,s)
      r:= r ++ [z];
      i:= i+1;
   return r
```

DummySum utility

```
DummySum(d : {0,1} list) : real list
   i:= 0;
   s:= 0;
   r:= [];
   while (i<size d)
      s:= s + d[i]
      z:=$ Lap(eps,s)
      r:= r ++ [z];
      i:= i+1;
   return r
```

What is the right metric here?

DummySum utility

```
DummySum(d : {0,1} list) : real list
   i:= 0;
   s:= 0;
   r:= [];
   while (i<size d)
      s:= s + d[i]
      z:=$ Lap(eps,s)
      r:= r ++ [z];
      i:= i+1;
   return r
```

What is the right metric here?

Can we use just the tail bound of Laplace?

Vector norms

We are releasing a real-valued vector, so we have several options to choose from. A meaningful choice could be l-infinity norm.

$$|f(x) - r|_\infty = \max_i |f(x)_i - r_i|$$

We could also consider other p-norms but the l-infinity norm gives us a natural worst case guarantee.

Union bound

A basic fact of probability theory gives us:

$$\Pr[A \cup B] \leq \Pr[A] + \Pr[B]$$

DummySum utility

```
DummySum(d : {0,1} list) : real list
   i:= 0;
   s:= 0;
   r:= [];
   while (i<size d)
      s:= s + d[i]
      z:=$ Lap(eps,s)
      r:= r ++ [z];
      i:= i+1;
   return r
```

We can show $\Pr_{r\sim DS(d)}[\,|f(d)-r|_\infty \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}] \leq \beta$

This follows as:

$$\Pr_{r\sim DP(d)}[\,|f(d)-r|_\infty \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}] = \Pr_{r\sim DP(d)}[\max_i |f(d)_i - r_i| \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}]$$

$$= \Pr_{r\sim DS(d)}[\{\,|f(d)_1 - r_1| \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}\} \cup \ldots \cup \{\,|f(d)_n - r_n| \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}\}]$$

$$\leq n \cdot \Pr_{r\sim Lap(f(d)_i)}[\,|f(d)_i - r| \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}] \leq n \cdot \frac{\beta}{n} = \beta$$

Report Noisy Max

```
RNM (q_1,...,q_N : (data → R) list,
     b : list data, ε: R) : nat
  i = 0;
  max = 0;
  while (i < N){
     cur = q_i(b) + Lap(1/ε)
     if (cur > max)
          max =  cur ;
          output = i;
  return output;
```

Can we use just the tail bound of Laplace and union bound?

Report Noisy Max

```
RNM (q_1,...,q_N : (data → R) list,
     b : list data, ε: R) : nat
  i = 0;
  max = 0;
  while (i < N){
     cur = q_i(b) + Lap(1/ε)
     if (cur > max)
          max =  cur ;
          output = i;
  return output;
```

Can we use just the tail bound of Laplace and union bound?

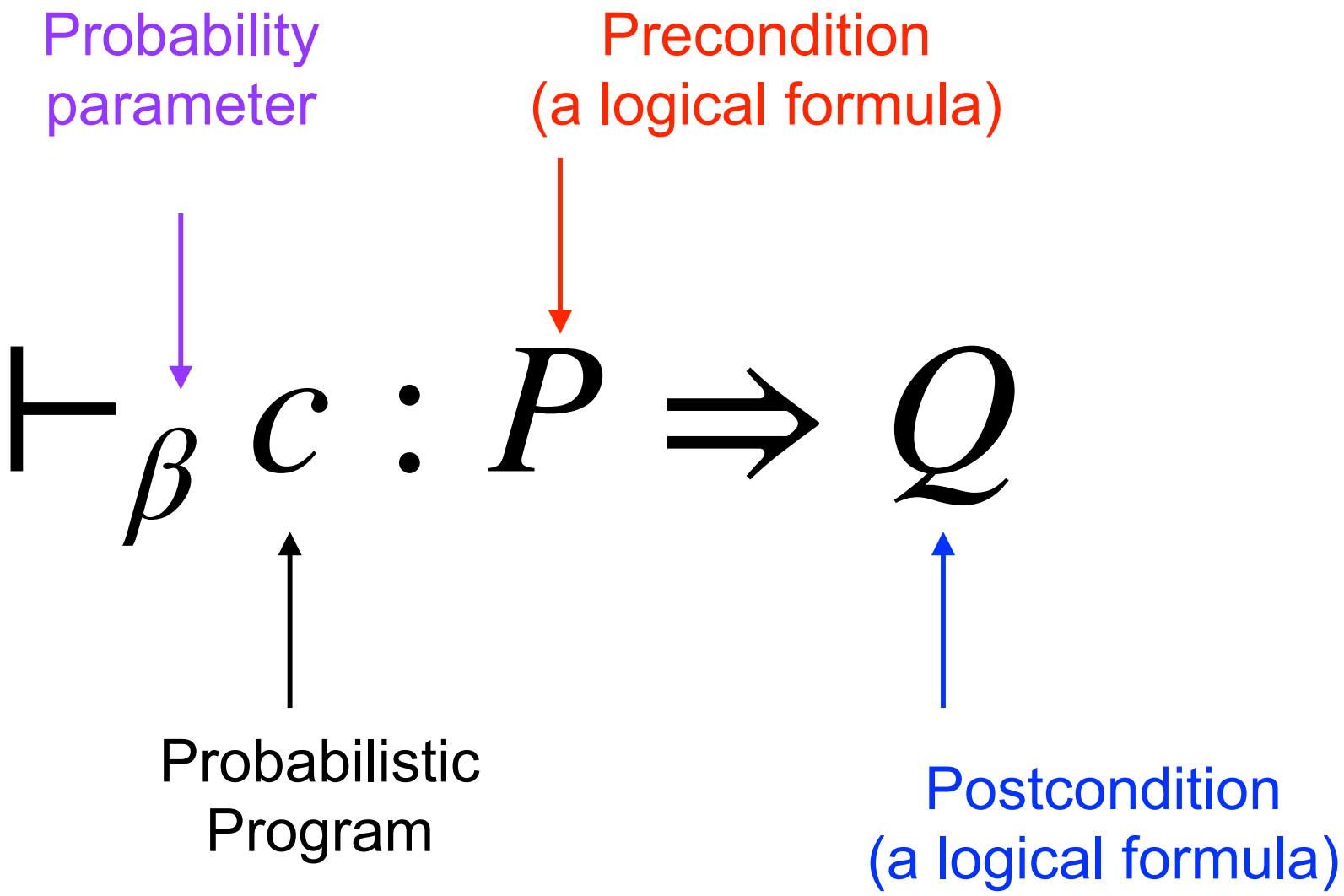
Yes, but the proof is more involved.

A logic for Union Bound

Just by using basic facts about Laplace and the union bound, one can prove utility statements for many DP mechanisms.

It also turns out that the union bound has an interesting counterpart in terms of logic….

uHL


Probability parameter
Precondition (a logical formula)
$\vdash_\beta c : P \Rightarrow Q$
Probabilistic Program
Postcondition (a logical formula)

Validity of uHL judgments

We say that the quadruple $\vdash_\beta c : P \Rightarrow Q$ is valid if and only if for every memory m such that $P(m)$ we have:
$\{c\}_m = \mu$ implies $Q^u_\beta(\mu)$.

Unary u-lifting

We say that a subdistribution $\mu \subseteq D(A)$ is in the unary u-lifting of the predicate $Q \subseteq A$, denoted $Q^{u}_{\beta}(\mu)$ if and only if we have:

$$Pr_{r \sim \mu}[r \notin Q] \leq \beta$$

Rules of uHL: Skip

$$\frac{}{\vdash_0 \mathtt{skip} : P \Rightarrow P}$$

Rules of uHL: Skip

$$\frac{}{\vdash_0 \mathtt{skip} : P \Rightarrow P}$$

Is this correct?

Rules of uHL: Assignment

$$\frac{}{\vdash_0 x := e \;:\; P[e/x] \Rightarrow P}$$

Rules of uHL: Assignment

$$\vdash_0 x := e \;:\; P[e/x] \Rightarrow P$$

Is this correct?

Rules of uHL
Composition

$$\frac{\vdash_{\beta} c : P \Rightarrow R \qquad \vdash_{\beta'} c' : R \Rightarrow Q}{\vdash_{\beta+\beta'} c ; c' : P \Rightarrow Q}$$

Rules of uHL
Composition

$$\frac{\vdash_{\beta} C : P \Rightarrow R \qquad \vdash_{\beta'} C' : R \Rightarrow Q}{\vdash_{\beta+\beta'} C ; C' : P \Rightarrow Q}$$

Is this correct?

Rules of uHL
Composition

$$\frac{\vdash_{\beta} c : P \Rightarrow R \qquad \vdash_{\beta'} c' : R \Rightarrow Q}{\vdash_{\beta+\beta'} c ; c' : P \Rightarrow Q}$$

Is this correct?

Correctness shows that this rule internalized the union bound.

Rules of uHL
If then else

$$\frac{\vdash_\beta c_1 : e \wedge P \Rightarrow Q \quad \vdash_\beta c_2 : \neg e \wedge P \Rightarrow Q}{\vdash_\beta \texttt{if } e \texttt{ then } c_1 \texttt{ else } c_2 : P \Rightarrow Q}$$

Is this correct?

Rules of uHL: Rand

$$\frac{\forall m, \mathbb{P}(m) \Rightarrow \mathbb{Q}^{u}\beta([x:=\$e]_{m})}{\vdash_{\beta} x:=\$e \ : \ \mathbb{P} \Rightarrow \mathbb{Q}}$$

Rules of uHL: Rand

$$\frac{\forall m, \mathbb{P}(m) \Rightarrow \mathbb{Q}^{u}_{\beta}([x :=\$e]_{m})}{\vdash_{\beta} x :=\$e \ : \ \mathbb{P} \Rightarrow \mathbb{Q}}$$

Is this correct?

Rules of uHL: Rand

$$\frac{\forall m, \mathbb{P}(m) \Rightarrow Q^{u}_{\beta}([x:=\$e]_{m})}{\vdash_{\beta} x:=\$e \;:\; \mathbb{P} \Rightarrow Q}$$

Is this correct?

Is this useful?

Rules of uHL
While

$$\frac{\forall n \vdash_0 c : P \wedge e \wedge v{=}n \Rightarrow v{<}n \qquad P \wedge v{\leq}0 \Rightarrow \neg e \qquad \vdash_\beta c : P \Rightarrow P}{\vdash_{k*\beta} \texttt{while } e \texttt{ do } c : P \wedge v{\leq}k \Rightarrow P \wedge \neg e}$$

Laplace Mechanism utility

```
Lap(d : priv data)(f: data -> real)
    (e:real) : pub real
   z:=$ Laplace(GS_f/e,0)
   z:= f(d)+z
   return z
```

We can show

$$\vdash_{\beta} \texttt{Lap:P} \Rightarrow |\texttt{f(d)}-\texttt{z}| \geq \ln(1/\beta)\,\texttt{GS}_\texttt{f}/\texttt{e}$$

Which corresponds to what we want

$$\Pr_{r \sim Lap(x)}[\,|f(x) - r| \geq \ln(\frac{1}{\beta})\frac{GS_f}{\epsilon}] \leq \beta$$

Mean Utility

```
Mean d : ({0,1} list) : real
   i:=0;
   s:=0;
   n:=size(d);
   while (i<n)
      s:=s + d[i]
      i:=i+1;
   z:=$ Laplace(1/(n*eps),0)
   z:= (s/n)+z
   return z
```

We can show

$$\vdash_{\beta} \texttt{Mean:T} \Rightarrow |\texttt{s/n-z}| \geq \texttt{ln(1/}\beta\texttt{)1/ne}$$

DummySum utility

```
DummySum(d : {0,1} list) : real list
   i:= 0;
   s:= 0;
   r:= [];
   while (i<size d)
      s:= s + d[i]
      z:=$ Lap(eps,s)
      r:= r ++ [z];
      i:= i+1;
   return r
```

We can show

$$\vdash_{\beta} \texttt{DS:T} \Rightarrow |\texttt{s-r}|_{\infty} \geq \texttt{ln}(|\texttt{d}|/\beta)\,\texttt{1/e}$$

Which corresponds to what we want:

$$\Pr_{r \sim DS(d)}[\,|f(d) - r|_{\infty} \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}] \leq \beta$$

This uses the while rule for the repeated union bound.

Soundness

If we can derive $\vdash_\beta c \ : \ P \Rightarrow Q$ through the rules of the logic, then the triple

$\vdash_\beta c \ : \ P \Rightarrow Q$ is valid.

Are the rules we presented complete?

NoisySum utility

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

NoisySum utility

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

Can we use just the tail bound of Laplace and the union bound?

NoisySum utility

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

Can we use just the tail bound of Laplace and the union bound?

$$\Pr_{r\sim DS(d)}[\,|f(d)-r|_\infty \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}] \leq \beta$$

NoisySum utility

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

Can we use just the tail bound of Laplace and the union bound?

$$\Pr_{r \sim DS(d)}[\,|f(d) - r|_\infty \geq \ln(\frac{n}{\beta})\frac{1}{\epsilon}] \leq \beta$$

Can we do better?

Independence of random variables

A set of random variables $X_1 \ldots X_n$ are independent if and only if

$$\Pr[X_1=x_1,\ldots,X_n=x_n]=\Pr[X_1=x_1]\Pr[X_2=x_2]\ldots\Pr[X_n=x_n]$$

for every $x_1,\ldots,x_n$

Concentration bounds

Under the assumption of independence for n random variables we can do better than the union bound.

For example Hoeffding's inequality gives a bound on the distance between the sum S_n of n independent boolean variables and the sum E_n of their expectations:

$$\Pr[\,|S_n - E[S_n]| > t] \leq 2\exp\left(\frac{2t^2}{n}\right)$$

NoisySum utility

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

We can prove

NoisySum utility

```
NoisySum(d : {0,1} list) : real list
   i:=0;
   s:=0;
   r:=[];
   while (i<size d)
      z:=$ Lap(eps,d[i])
      s:= s + z
      r:= r ++ [s];
      i:= i+1;
   return r
```

We can prove

$$\Pr_{r \sim DS(d)}[\,|f(d) - r|_\infty \geq \ln(\frac{\sqrt{n}}{\sqrt{2\beta}})\frac{1}{\epsilon}] \leq \beta$$

A logic for independence

- We have seen that to have improved probability bounds for sums we can use independence,
- Can we design a simple logic for independence?

Judgments

$$\{\xi\} \; s \; \{\xi'\}$$

Assertions

$$\xi \quad := \quad \mathsf{det}(e) \mid \#\, E \mid e \sim \mathrm{B}(e, p) \mid \top \mid \bot \mid \xi \wedge \xi$$

Assertions

$$\xi \quad := \quad \det(e) \mid \#\, E \mid e \sim \mathrm{B}(e, p) \mid \top \mid \bot \mid \xi \wedge \xi$$

At most one element in the support of e

Assertions

The expressions in E are independent

$$\xi \quad := \quad \det(e) \mid \#E \mid e \sim \mathrm{B}(e, p) \mid \top \mid \bot \mid \xi \wedge \xi$$

At most one element in the support of e

Assertions

The expressions in E are independent

$$\xi \quad := \quad \det(e) \mid \#E \mid e \sim \mathrm{B}(e, p) \mid \top \mid \bot \mid \xi \wedge \xi$$

At most one element in the support of e

e is distributed as a Binomial

Some axioms

$$\#\{x\} \qquad \#\{x,x\} \iff \mathsf{det}(x) \qquad \#(E \cup E') \implies \#\, E$$

$$e \sim \mathrm{B}(m,p) \wedge e' \sim \mathrm{B}(m',p) \wedge \#\{e,e'\} \implies e+e' \sim \mathrm{B}(m+m',p)$$

$$\bigwedge_{1 \le i \le n} \mathsf{det}(e_i) \implies \mathsf{det}(f(e_1,\ldots,e_n))$$

Some rules

$$\frac{\{x\} \cap \mathrm{FV}(E) \cap \mathrm{FV}(e) = \emptyset}{\{\#\, E\}\ x \xleftarrow{\$} \mathrm{B}(e,p)\ \{\#(E \cup \{x\}) \wedge x \sim \mathrm{B}(e,p)\}}$$

Some rules

$$\frac{\{\xi\}\ s_1\ \{\xi'\} \qquad \{\xi\}\ s_2\ \{\xi'\} \qquad \xi \implies \mathsf{det}(b)}{\{\xi\}\ \mathbf{if}\ b\ \mathbf{then}\ s_1\ \mathbf{else}\ s_2\ \{\xi'\}}$$

Some rules

$$\frac{\{\xi\}\ s\ \{\xi\} \qquad \xi \implies \mathsf{det}(b) \qquad \mathcal{C}_{\mathrm{CTerm}}}{\{\xi\}\ \mathbf{while}\ b\ \mathbf{do}\ s\ \{\xi\}}$$

Proving the accuracy of NoisySum

- The logic of independence we presented doesn’t express bounds — nor failure probabilities,
- to prove accuracy for NoisySum we need to combine it with the union bound logic,
- one option is to embed both

Implementations

- The logic for union bound has been implemented as a prototype in EasyCrypt,
- The logic for independence has been implemented using Ellora, a probabilistic program logic using probabilistic assertions.