Higher-Order Relational Refinement Types for Mechanism Design and Differential Privacy

Gilles Barthe\(^1\), Marco Gaboardi\(^2\), Emilio Jesús Gallego Arias\(^3,4\), Justin Hsu\(^4\), Aaron Roth\(^4\), Pierre-Yves Strub\(^1\)

\(^1\)IMDEA Software, \(^2\)University of Dundee, \(^3\)CRI Mines–ParisTech, \(^4\)University of Pennsylvania

January 15th, 2015
The Application

Mechanism Design
One painting for sale

$10 million!

$50 million!

$10.1 million?

Who wins, and for how much?
One painting for sale

How much will you pay?

$10 million!

$50 million!

$10.1 million?

Who wins, and for how much?
A story

One painting for sale

How much will you pay?

$10 million!

$50 million!

$3
One painting for sale

How much will you pay?

$10 million!

$50 million!

$3

Who wins, and for how much?
Top bid pays top price?

- Simple rule
- Can encourage manipulation...

How much will you pay?

- $10 million!
- $50 million!
- $3
A story

How much will you pay?

Top bid pays top price?

- Simple rule
- Can encourage manipulation...

$10 million!

$50 million!

$10.1 million?

$3
Algorithm design with strategic inputs
What is Mechanism Design?

Algorithm design with strategic inputs

Rational agents
- Report data
- Care about output
- May lie, strategize
What is Mechanism Design?

Algorithm design with strategic inputs

Rational agents

- Report data
- Care about output
- May lie, strategize

Goal: encourage “good” behavior
Designing auctions

- Bidders each have personal value \( v : \mathbb{R} \) for the item
Designing auctions

- Bidders each have personal value $v : \mathbb{R}$ for the item
- Bidder’s happiness is function of price, $v$, whether they win
Designing auctions

- Bidders each have personal value $v : \mathbb{R}$ for the item
- Bidder’s happiness is function of price, $v$, whether they win
- Bidder reports a bid $b : \mathbb{R}$ to the mechanism
Designing auctions

- Bidders each have personal value $v : \mathbb{R}$ for the item
- Bidder’s happiness is function of price, $v$, whether they win
- Bidder reports a bid $b : \mathbb{R}$ to the mechanism

Property: agent always maximizes happiness with $b = v$
Fixed price auction

- Given a fixed price \( \text{price} \)
- Bidder bids \( \text{bid} \), buys item if higher than \( \text{price} \)

What is the happiness function for a bidder?

\[
\text{fixedprice price value bid} = \begin{cases} 
\text{value} - \text{price} & \text{if } \text{bid} > \text{price} \\
0 & \text{else}
\end{cases}
\]
A (very) simple auction

Fixed price auction

- Given a fixed price \( \text{price} \)
- Bidder bids \( \text{bid} \), buys item if higher than \( \text{price} \)

What is the happiness function for a bidder?

\[
\text{fixedprice price value bid} = \\
\text{if } \text{bid} > \text{price} \text{ then } \\
\quad \text{value} - \text{price} \\
\text{else } \\
\quad 0
\]
Consider bidder’s happiness function…

- First run: bidder bids $b = v$ (honest)
Consider bidder’s happiness function…

- First run: bidder bids $b = v$ (honest)
- Second run: bidder bids arbitrarily (maybe not honest)
Consider bidder’s happiness function…

- First run: bidder bids \( b = v \) (honest)
- Second run: bidder bids arbitrarily (maybe not honest)
- Verify: happiness in first run is higher than in second run
Consider bidder’s happiness function…

- First run: bidder bids $b = v$ (honest)
- Second run: bidder bids arbitrarily (maybe not honest)
- Verify: happiness in first run is higher than in second run
Consider bidder’s happiness function...

- First run: bidder bids $b = v$ (honest)
- Second run: bidder bids arbitrarily (maybe not honest)
- Verify: happiness in first run is higher than in second run

\[
\text{fixedprice } p \ v \ v = \begin{cases} 
  v - p & \text{if } v > p \\
  0 & \text{else}
\end{cases}
\]

\[
\text{fixedprice } p \ v \ b = \begin{cases} 
  v - p & \text{if } b > p \\
  0 & \text{else}
\end{cases}
\]
Consider bidder’s happiness function…

- First run: bidder bids \( b = v \) (honest)
- Second run: bidder bids arbitrarily (maybe not honest)
- Verify: happiness in first run is higher than in second run

\[
\text{fixedprice } p \ v \ v = \\
\text{if } v > p \text{ then } v - p \\
\text{else } 0
\]

\[
\text{fixedprice } p \ v \ b = \\
\text{if } b > p \text{ then } v - p \\
\text{else } 0
\]

This is a relational property
Introducing \textsc{HOARRe}\textsuperscript{2}

A type system with relational refinement types
Judgment

$$\Gamma \vdash e : \{ x : T \mid \phi(x) \}$$
Refinement types

**Judgment**

\[ \Gamma \vdash e : \{ x : T \mid \phi(x) \} \]
Refinement types

Judgment

\[ \Gamma \vdash e : \{ x : T \mid \phi(x) \} \]
Judgment

\( \Gamma \vdash e : \{ x : T \mid \phi(x) \} \)

“\( e \) is a program of type \( T \) such that \( \phi(e) \) holds”
Example

\[ \Gamma \vdash 3 : \{ x : \mathbb{Z} \mid x \geq 0 \} \]
Example

$$\Gamma \vdash 3 : \{x : \mathbb{Z} \mid x \geq 0\}$$

"3 is a non-negative integer"
Relational Reasoning

Relational Judgment

\[ \Gamma \vdash e :: \{ x :: T \mid \phi( x_{\le}, x_{\ge} ) \} \]
Relational Reasoning

Relational Judgment

\[ \Gamma \vdash e :: \{ x :: T \mid \phi(x_{\sqsubseteq}, x_{\triangleright}) \} \]
Relational Reasoning

Relational Judgment

\[ \Gamma \vdash e :: \{ x :: T \mid \phi(x_{\leftarrow}, x_{\rightarrow}) \} \]

\( \phi \) mentions two runs of program \( e \) via \( x_{\leftarrow} \) and \( x_{\rightarrow} \)
Relational Judgment

\[ \Gamma \vdash e :: \{ x :: T \mid \phi(x_{\downarrow}, x_{\uparrow}) \} \]

\( \phi \) mentions two runs of program \( e \) via \( x_{\downarrow} \) and \( x_{\uparrow} \)

Example

\[ \{ y :: \mathbb{Z} \mid y_{\downarrow} \leq y_{\uparrow} \} \vdash e :: \{ x :: \mathbb{Z} \mid x_{\downarrow} \leq x_{\uparrow} \} \]
Relational Judgment

$$\Gamma \vdash e :: \{ x :: T \mid \phi(x_\triangleleft, x_\triangleright) \}$$

$\phi$ mentions two runs of program $e$ via $x_\triangleleft$ and $x_\triangleright$

Example

$$\{ y :: \mathbb{Z} \mid y_\triangleleft \leq y_\triangleright \} \vdash e :: \{ x :: \mathbb{Z} \mid x_\triangleleft \leq x_\triangleright \}$$

"If $y$ increases, then $e$ increases."
Relational Judgment

\[ \Gamma \vdash e :: \{ x :: T \mid \phi(x_{\triangleleft}, x_{\triangleright}) \} \]

\( \phi \) mentions two runs of program \( e \) via \( x_{\triangleleft} \) and \( x_{\triangleright} \)

Example

\[ \{ y :: \mathbb{Z} \mid y_{\triangleleft} \leq y_{\triangleright} \} \vdash e :: \{ x :: \mathbb{Z} \mid x_{\triangleleft} \leq x_{\triangleright} \} \]

“\( y \) increases, then \( e \) increases.”

Background

- First used in the RF* language, POPL 2014
Happiness function

```plaintext
fixedprice price value bid =
  if bid > price then
    value - price
  else
    0
```
Happiness function

\[
\text{fixedprice } price \text{ value bid } = \\
\text{if } bid > price \text{ then}
\text{value - price}
\text{else}
0
\]

Truthfulness in a type
Happiness function

\[
\text{fixedprice \ price \ value \ bid =}
\begin{align*}
\text{if bid > price then} \\
\text{value - price} \\
\text{else} \\
0
\end{align*}
\]

Truthfulness in a type

\[
\{ p :: \mathbb{R} \mid p_\downarrow = p_\uparrow \} \quad \text{(Fixed price)}
\]
Happiness function

\[
\text{fixedprice price value bid} = \\
\begin{cases} 
\text{value} - \text{price} & \text{if bid} > \text{price} \\
0 & \text{else}
\end{cases}
\]

Truthfulness in a type

\[
\{ p :: \mathbb{R} \mid p_\downarrow = p_\uparrow \} 
\rightarrow \{ v :: \mathbb{R} \mid v_\downarrow = v_\uparrow \}
\]

(Fixed price)

(Bidder value fixed)
Typing truthfulness

Happiness function

fixedprice price value bid =
if bid > price then
  value - price
else
  0

Truthfulness in a type

\{ p :: \mathbb{R} \mid p_{\downarrow} = p_{\uparrow} \} \quad \text{(Fixed price)}
\rightarrow \{ v :: \mathbb{R} \mid v_{\downarrow} = v_{\uparrow} \} \quad \text{(Bidder value fixed)}
\rightarrow \{ b :: \mathbb{R} \mid b_{\downarrow} = v_{\downarrow} \} \quad \text{(Bid = value on \triangleleft run)}
### Happiness function

```plaintext
fixedprice price value bid =
  if bid > price then
    value - price
  else
    0
```

### Truthfulness in a type

- \( p :: \mathbb{R} \mid p_{\downarrow} = p_{\uparrow} \)  
  - (Fixed price)
- \( v :: \mathbb{R} \mid v_{\downarrow} = v_{\uparrow} \)  
  - (Bidder value fixed)
- \( b :: \mathbb{R} \mid b_{\downarrow} = v_{\downarrow} \)  
  - (Bid = value on \( \downarrow \) run)
- \( u :: \mathbb{R} \mid u_{\downarrow} \geq u_{\uparrow} \)  
  - (Truthful)
A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?
Adding in randomness

A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?

Randomize!
A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?

Randomize!
A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?

Randomize!

Verify: Happiness higher when bid is the true value on average.
A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?

Randomize!
Adding in randomness

A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?

Randomize!

Verify: happiness higher when bid is true value
A more complex auction

- Unlimited supply of items (e.g., music files)
- Want to use fixed price, but for what price?

Randomize!

Verify: happiness higher when bid is true value
Monotonicity of expectation

- (One) Distribution $\mu$ over $A$
Monotonicity of expectation

- (One) Distribution $\mu$ over $A$
- Two functions $f_1, f_2 : A \to \mathbb{R}$ with
  $$f_1(x) \geq f_2(x) \quad \text{for all } x : A$$
Monotonicity of expectation

- (One) Distribution $\mu$ over $A$
- Two functions $f_1, f_2 : A \rightarrow \mathbb{R}$ with

  \[ f_1(x) \geq f_2(x) \quad \text{for all } x : A \]

- Then, fact about expected values:

  \[ E_\mu[f_1] \geq E_\mu[f_2] \]
Monotonicity of expectation

- (One) Distribution $\mu$ over $A$
- Two functions $f_1, f_2 : A \to \mathbb{R}$ with

$$f_1(x) \geq f_2(x) \quad \text{for all } x : A$$

- Then, fact about expected values:

$$E_\mu[f_1] \geq E_\mu[f_2]$$

$f_1$ bigger than $f_2$ on average
Extending HOARe$^2$

Distributions and Higher-order refinements
Probabilistic programs

- Reason about two runs of a probabilistic program
- Use type of probability distributions
Probabilistic programs

- Reason about two runs of a probabilistic program
- Use type of probability distributions

Typing distributions

$$\Gamma \vdash e :: M_{0,0}[\{x :: T \mid \phi(x_{\sqsubseteq}, x_{\sqsupset})\}]$$
Probabilistic programs

- Reason about two runs of a probabilistic program
- Use type of probability distributions

Typing distributions

\[
\Gamma \vdash e :: M_{0,0}[\{x :: T \mid \phi(x_\ll, x_\gg)\}]
\]

“\(e\) is a distribution over \(T\), with two runs related by \(\phi\)”
Probabilistic programs

- Reason about two runs of a probabilistic program
- Use type of probability distributions

Typing distributions

\[ \Gamma \vdash e :: M_{0,0}[\{x :: T \mid \phi(x_\triangleleft, x_\triangleright)\}] \]

"e is a distribution over \( T \), with two runs related by \( \phi \)"
What does this mean?

- Convert relation \( \phi \) to a relation \( \phi^\# \) on distributions over \( T \)
- Two runs of \( e \) related by \( \phi^\# \) (as distributions!)
Example

\[ \Gamma \vdash e :: M_{0,0}[\{x :: T \mid x_{\triangleleft} = x_{\triangleup}\}] \]
Example

\[ \Gamma \vdash e :: M_{0,0} \left[ \{ x :: T \mid x_{\triangleleft} = x_{\triangleright} \} \right] \]
Example

\[ \Gamma \vdash e :: M_{0,0}[\{x :: T \mid x_\triangleleft = x_\triangleright}\} ] \]

"e is a distribution over T that is identical in both runs"
Equivalence of Distributions

Example

\[
\Gamma \vdash e :: M_{0,0}\left[\{x :: T \mid x_\downarrow = x_\uparrow\}\right]
\]

“\(e\) is a distribution over \(T\) that is identical in both runs”

Background

- Proposed by Barthe, Köpf, Olmedo, Zanella
- Generalizing 0, 0 to \(\varepsilon, \delta\) models differential privacy
Example

\[ \Gamma \vdash e :: M_{0,0}[\{x :: T \mid x_\downarrow = x_\uparrow\}] \]

"e is a distribution over T that is identical in both runs"

Background

- Proposed by Barthe, Köpf, Olmedo, Zanella
- Generalizing 0, 0 to \( \varepsilon, \delta \) models differential privacy

Our contribution

- Simplify and build into a type system
Higher-Order Refinements

Refinements on functions

\[ \Gamma \vdash e :: \{ f :: T \to U \mid \phi \} \]
Refinements on functions

\[ \Gamma \vdash e :: \{ f :: T \rightarrow U \mid \phi \} \]

“\(e\) is a function from \(T\) to \(U\) that satisfies \(\phi\)”
Higher-Order Refinements

Refinements on functions

\[ \Gamma \vdash e :: \{ f :: T \rightarrow U \mid \phi \} \]

“\(e\) is a function from \(T\) to \(U\) that satisfies \(\phi\)”

Our contribution

- Consistency by carefully handling termination
Refinements on functions

\[ \Gamma \vdash e :: \{ f :: T \to U \mid \phi \} \]

“\( e \) is a function from \( T \) to \( U \) that satisfies \( \phi \)”

Our contribution

- Consistency by carefully handling termination
- Show naïve treatment leads to inconsistency
Want to show

\[ \mathbb{E} \mu f_1 \geq \mathbb{E} \mu f_2 \]

In HOARe\(^2\), type \(\mathbb{E}\) as...

\[ \mathcal{M}_{0,0}\{x :: A \mid x_\sqsubseteq = x_\triangleright\} \] (Same distributions)
Expressing monotonicity of expectations

Want to show

\[ \mathbb{E} \mu f_1 \geq \mathbb{E} \mu f_2 \]

In HOARê^2, type \( \mathbb{E} \) as...

\[ \mathcal{M}_{0,0}[\{x :: A \mid x_{\triangleleft} = x_{\triangleright}\}] \] \hfill (Same distributions)

\[ \rightarrow \{f :: A \rightarrow \mathbb{R} \mid \forall x. f_{\triangleleft} x \geq f_{\triangleright} x\} \] \hfill (Higher-order)
Expressing monotonicity of expectations

Want to show

\[ \mathbb{E} \mu f_1 \geq \mathbb{E} \mu f_2 \]

In HOAR\textsuperscript{e2}, type $\mathbb{E}$ as ...

\[\mathcal{M}_{0,0}[\{x :: A \mid x_{\downarrow} = x_{\uparrow}\}] \quad \text{(Same distributions)}\]
\[\rightarrow \{ f :: A \rightarrow \mathbb{R} \mid \forall x. f_{\downarrow} x \geq f_{\uparrow} x \} \quad \text{(Higher-order)}\]
\[\rightarrow \{ e :: \mathbb{R} \mid e_{\downarrow} \geq e_{\uparrow} \} \quad \text{(Monotonic)}\]
Expressing monotonicity of expectations

Want to show

\[ \mathbb{E} \mu f_1 \geq \mathbb{E} \mu f_2 \]

In HOARRe², type \( \mathbb{E} \) as...

\[ \mathbb{E} :: \mathcal{M}_{0,0}[\{x :: A \mid x_\triangleleft = x_\triangledown\}] \]  
\[ \rightarrow \{f :: A \rightarrow \mathbb{R} \mid \forall x. f_\triangleleft x \geq f_\triangledown x\} \]  
\[ \quad \text{(Same distributions)} \]
\[ \rightarrow \{e :: \mathbb{R} \mid e_\triangleleft \geq e_\triangledown\} \]  
\[ \quad \text{(Higher-order)} \]
\[ \rightarrow \{e :: \mathbb{R} \mid e_\triangleleft \geq e_\triangledown\} \]  
\[ \quad \text{(Monotonic)} \]
Semantics
- Soundness of the system
- Requires termination

Implementation
- Automated, low annotation burden
- Why3 and SMT solvers

Translation
- Embedding of DFuzz, a language for differential privacy

More complex examples
- Verify differential privacy
- Verify MD properties beyond truthfulness
Takeaway points
Four features, one system

- HOARe\textsuperscript{2}: relational properties for randomized programs
- Combine features in a clean, usable way
Four features, one system

- HOARe²: relational properties for randomized programs
- Combine features in a clean, usable way

Formal verification for mechanism design!

- Exciting, under-explored area for verification
- Tons of interesting properties, mechanisms
- Strong motivation besides (mere) correctness
Higher-Order Relational Refinement Types for Mechanism Design and Differential Privacy

Gilles Barthe¹, Marco Gaboardi², Emilio Jesús Gallego Arias³,⁴, Justin Hsu⁴, Aaron Roth⁴, Pierre-Yves Strub¹

¹IMDEA Software, ²University of Dundee, ³CRI Mines–ParisTech, ⁴University of Pennsylvania

January 15th, 2015