A new approach to program distance for HO sequential languages

A foundation for approximate program transformations

First step towards a differential program semantics

Let us proceed step by step...
From Program Equivalence to Program Distance
What is program distance?

A natural refinement of program equivalence: do programs have the same behavior?

\[
\begin{align*}
\left( \begin{array}{c}
\text{let } x_1 = e_1 \\
\text{let } x_2 = e_2 \\
\text{in } e
\end{array} \right) & \equiv \\
\left( \begin{array}{c}
\text{let } x_2 = e_2 \\
\text{let } x_1 = e_1 \\
\text{in } e
\end{array} \right)
\end{align*}
\]

Useful for many applications:

- Verification
- Security
- Optimization
Program Equivalence

Key points:

(1) Same observable behaviour:

\[ e \equiv f \implies \text{Obs}(e) = \text{Obs}(f) \]

(2) Same interaction with the environment:

\[ \forall C. \, e \equiv f \implies C[e] \equiv C[f] \]

Compositionality
Program equivalence works fine for **exact** program analysis

Too strong for **quantitative** and **approximate** program analysis

- Probabilistic programming
- Differential privacy
- Real-valued computing
- **Approximate computing**
The Challenge of Approximate Computing

[Mittal 2016] Introducing soft errors in exact computations can provide disproportionate gains in efficiency and resource-consumption.

Many applications:
- Computer vision
- Sensor-based computation
- Data analytics and machine learning
The Challenge of Approximate Computing

[Misailovic et al. 2010] A language-based approach to approximate computing via approximate program transformation

Program transformation

\[ e \mapsto T(e) \]

**Correct:** \( e \) and \( T(e) \) have same output (\( e \equiv T(e) \))

\((\varepsilon-)\) **Approximately correct:** \( e \) and \( T(e) \) can have \((\varepsilon-)\) different outputs
The Challenge of Approximate Computing: Examples

Loop perforation [Misailovic et al. 2010;2011]: skip iterations in expensive loops over large dataset

$c$ skip factor:

\[
\text{for (i = 0; i < b; i++) } \{ e \} \quad \Downarrow T \quad \text{for (i = 0; i < b; i += c) } \{ e \}
\]

Example

\[
\left( \text{for (i = 0; i < N; i++) } \begin{array}{c}
\text{sum += A[i]}
\end{array} \right) \quad T \quad \rightarrow \quad \left( \text{for (i = 0; i < N; i+=k) } \begin{array}{c}
\text{sum += A[i]}
\end{array} \right)
\]
Numerical integrations [Zeyuan et al. 2012]: $e$ integrates a function $F : \mathbb{R} \to \mathbb{R}$ over $[a, b]$

let $\Delta = \frac{b-a}{n}$

$x_i = a + i \cdot \frac{\Delta}{2}$

in

$$\frac{1}{n} \sum_{i=1}^{n} (b - a) \cdot F(x_i)$$

E.g. $F(x) = x \cdot \sin(\log(x))$
The Challenge of Approximate Computing: Examples

**Substitution:** Substitute $F$ with $F'$ computing a less accurate output in less time

**Perforation:** Instead of computing $F(x_i)$ for all $n$-inputs, take only $s \leq n$ inputs
Approximate program transformations raise several questions

How to reason about APTs? When can we safely apply them?

What about compositionality?

The classic theory of program equivalence is useless...
Program Distance
Obviously, $e \not\equiv T(e) \ldots$ but differences bounded by accuracy $\varepsilon$

Natural guess: refine equivalences into distances

[Lawvere 1973]: equivalences and pseudometrics are essentially the same
Program Distance

What about compositionality?

Program Equivalence: compositionality = congruence

\[ \forall C. \ e \equiv f \implies C[e] \equiv C[f] \]

Program distance: compositionality = non-expansiveness

\[ \delta(e, f) \geq \sup_C \delta(C[e], C[f]) \]

Question: is this the end of the story? Not for higher-order languages...
A Toy Language and its Denotational Semantics

We fix a simply-typed $\lambda$-calculus with primitives for real numbers

$$\tau ::= \text{real} \mid \tau \rightarrow \tau \mid \tau \times \tau$$

Higher-order language

$$\frac{\Gamma, x : \tau \vdash e : \sigma}{\Gamma \vdash \lambda x.e : \tau \rightarrow \sigma} \quad \frac{\Gamma \vdash f : \tau \rightarrow \sigma \quad \Gamma \vdash e : \tau}{\Gamma \vdash fe : \sigma}$$

Real-valued computations

$$\frac{\Gamma \vdash r : \text{real}}{\Gamma \vdash e_i : \text{real} \quad F : \mathbb{R}^n \rightarrow \mathbb{R}}{\Gamma \vdash F(e_1, \ldots, e_n) : \text{real}}$$

Standard denotational semantics

$$[\text{Real}] = \mathbb{R} \quad [\tau \rightarrow \sigma] = [\tau] \rightarrow [\sigma] \quad [\tau \times \sigma] = [\tau] \times [\sigma]$$
Distance Amplification

Theorem [Crubillé & Dal Lago 2017; Gavazzo 2019]

Suppose:

1. \( \delta : \Lambda \times \Lambda \rightarrow [0, \infty] \) pseudometric
2. \( \delta \) is non-expansive
3. \( \delta \) is adequate: \( \forall e, f : \text{real}. \delta(e, f) \geq \|[e] - [f]\| \)

Then, \( \delta(e, f) > 0 \implies \delta(e, f) = \infty \)
Distance Amplification: Solutions

Several solutions . . .

- Denotationally-based distances [de Amorim et al. 2017]
- Tuple and contextual distance [Crubillé & Dal Lago 2017]
- Applicative distances [Gavazzo 2018]
- Metric logical relations [Reed & Pierce 2010]
Design a type system for program sensitivity (a.k.a. program robustness [Chaudhuri et al. 2011])

A program $e$ has sensitivity $c \in [0, \infty]$ if

$$\delta(v, v') \leq \varepsilon \implies \delta(e(v), e(v')) \leq c \cdot \varepsilon$$

Sensitivity tracked using a bounded linear type system

- $\sigma \rightarrow \tau$ non-expansive functions
- $!_c \sigma \rightarrow \tau$ $c$-Lipschitz continuous functions
Metric Logical Relations

Main Result

A metric logical relation inducing a **pseudometric** $\delta^L$ s.t.

- **Adequacy**
  \[ \delta^L_{\text{real}}(e, f) \geq \| [e] - [f] \| \]

- **Respects function space**
  \[ \delta^L_{\tau \rightarrow \sigma}(\lambda x . e, \lambda x . f) \geq \sup_v \delta^L_{\sigma}(e[v/x], f[v/x]) \]

- **Metric preservation** For any $\vdash e : !c \sigma \rightarrow \tau$
  \[ \delta^L_{\sigma}(v, v') \leq \varepsilon \implies \delta^L_{\tau}(e(v), e(v')) \leq c \cdot \varepsilon \]
Metric Logical Relations

Metric logical relations and sensitivity analysis applicable to

- Differential privacy [Reed & Pierce 2010; de Amorim et al. 2017; 2019]
- Probabilistic computations [Gavazzo 2018]
- Approximate computing
  Loop perforation [Chaudhuri et al. 2011]

Question: Is this really the end of the story? . . . Not really . . .
Recall our numerical integration example

```
let \( \Delta = \frac{b-a}{n} \)

\( x_i = a + i \cdot \frac{\Delta}{2} \)

in

\[ \frac{1}{n} \sum_{i=1}^{n} (b - a) \cdot F(x_i) \]
```

and the substitution technique: we substitute \( F \) with \( F' \) computing a less accurate output in less time
In approximate computing substitution is usually context-aware.

**Context aware-substitution**: Substitute $F$ with $F'$ computing a less accurate output in less time on inputs in a given set $A$.

If input $\in [0.5, 1]$

$$\lambda x. \frac{1}{x} \mapsto \lambda x. 2.823 - 1.882 \times x$$

If input $\approx 0$

$$\lambda x. \sin x \mapsto \lambda x. x$$
\[ e_{id} = \lambda x. x \]

\[ e_{\text{sin}} = \lambda x. \sin x \]
Can we use $\delta^L$ to prove approximate correctness of $T(e_{\sin}) = e_{id}$? No!

$$\delta^L(e_{id}, e_{\sin}) = \infty$$

Even worst . . .

**Proposition**

**For any pseudometric** $\delta : \Lambda \times \Lambda \rightarrow [0, \infty]$ which is **adequate** and respect function spaces

$$\delta_{\tau \circ \sigma}(\lambda x.e, \lambda x.f) \geq \sup_v \delta_{\sigma}(e[v/x], f[v/x])$$

we have

$$\delta(e_{id}, e_{\sin}) = \infty$$
Differential Logical Relations
A striking intuition

A Semantics for Approximate Program Transformations

Edwin Westbrook and Swarat Chaudhuri
Department of Computer Science, Rice University
Houston, TX 77005
Email: {emw4,swarat}@rice.edu

Thinking to program differences as just numbers is too restrictive.
Towards Differential Logical Relations

Replace \([0, \infty]\) in

\[\delta : \Lambda \times \Lambda \rightarrow [0, \infty]\]

with a more complex space of differences

\[\delta : \Lambda \times \Lambda \rightarrow \mathcal{D}\]

\(\mathcal{D}\) reflects the interactive complexity of programs, hence type-dependent
Question: What is a **difference** between two programs?

- Ordinary logical relations: boolean values.
- Metric logical relations: (real) numbers.
- Differential logical relations: **difference spaces** \(|\tau|\)
Differential Logical Relations

What is a difference between two (real) numbers?

\[ |\text{real}| = [0, \infty] \]

What is a difference between two functions \[ e, f : \sigma \rightarrow \tau \]?

\[ (\sigma \rightarrow \tau) = [\sigma] \times (|\sigma|) \rightarrow (|\tau|) \]

Intuition

\[ (\text{Input} \rightarrow \text{Output}) \approx \text{Input} \times \text{Error(Input)} \rightarrow \text{Error(Output)}. \]
Putting things together...

\[(\text{real}) = [0, \infty]\]

\[(\sigma \rightarrow \tau) = [[\sigma]] \times (\sigma) \rightarrow (\tau)\]

\[(\sigma \times \tau) = (\sigma) \times (\tau).\]
A differential logical relations is a ternary relation $\mathcal{D}_\tau \subseteq \Lambda_\tau \times (|\tau|) \times \Lambda_\tau$

**Informal reading**

$\mathcal{D}_\tau(e, \phi, f) \iff "\phi \text{ is a difference between } \vdash e, f : \tau"$

What are the defining clauses of $\mathcal{D}$?

- $\mathcal{D}_{\text{real}}(e, r, f) \iff |\llbracket e \rrbracket - \llbracket f \rrbracket| \leq r$
- $\mathcal{D}_{\tau_1 \times \tau_2}(e, (\phi_1, \phi_2), f) \iff \forall i \in \{1, 2\}. \mathcal{D}_{\tau_i}(e.i, \phi_i, f.i)$
- $\mathcal{D}_{\sigma \rightarrow \tau}(e, \phi, f) \iff ???$
How to define $D_{\sigma \rightarrow \tau}(e, \phi, f)$?

**Intuition**

$$\langle \text{Input} \rightarrow \text{Output} \rangle \approx \text{Input} \times \text{Diff(Input)} \rightarrow \text{Diff(Output)}.$$ 

Hence $\phi : [\sigma] \times (\sigma) \rightarrow (\tau)$

\[
\begin{align*}
D_{\sigma \rightarrow \tau}(e, \phi, f) \iff & \forall v, w \in V_\sigma. \\
& \forall \text{ inputs} \\
\forall \zeta \in (\sigma). \ D_\sigma(v, \zeta, w) \implies & \begin{cases} 
D_\tau(ev, \phi([v], \zeta), fw) \\
D_\tau(ew, \phi([v], \zeta), fv) \\
\phi([v], \zeta) \text{ output difference}
\end{cases}
\end{align*}
\]
How can we use DLRs to reason about context-aware approximate program transformation?

Compositional reasoning for ordinary and metric logical relations follows from fundamental lemma.

What is FL for differential logical relations?

<table>
<thead>
<tr>
<th>Logical Relations</th>
<th>Fundamental Lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ordinary</td>
<td>$R(e, e) = true$</td>
</tr>
<tr>
<td>Metric</td>
<td>$\delta(e, e) = 0$</td>
</tr>
<tr>
<td>Differential</td>
<td>???</td>
</tr>
</tbody>
</table>
Differential Logical Relations

Fundamental Lemma
For any ⊢ e : τ, there exists φ ∈ (|τ|) such that \( D_\tau(e, \phi, e) \).

Why non-null differences?

- \( \phi \) describes the sensitivity of \( e \).
- E.g. \( \vdash \lambda x.x : \text{real} \rightarrow \text{real} \) has self-difference
  \[
  \lambda (r, \varepsilon). \varepsilon
  \]
- Thus only terms at ground types have null-difference.
Fundamental Lemma $\implies$ compositional reasoning

- Given a context: $C : \sigma \rightarrow \tau$
- FL $\implies \exists \zeta \in (\sigma \rightarrow \tau). (C, \zeta, C') \in D_{\sigma \rightarrow \tau}$
- Given terms $\vdash e, f : \sigma$ s.t. $(e, \phi, f) \in D_{\sigma}$
- Conclude:
  
  $$(C[e], \zeta([e], \phi), C[f]) \in D_{\tau}$$

Moral: $\zeta$ captures context-awareness

The impact of replacing $f$ with $e$ in $C$ is $\zeta([e], \phi)$
Look back at $e_{id}$ vs $e_{\sin}$

Lemma

$$(e_{id}, \lambda (r, \varepsilon). \varepsilon + |\sin r - r|, e_{\sin}) \in D_{\text{real} \rightarrow \text{real}}$$

Consider the context $\vdash C = (\lambda x..x(xc))[\varepsilon] : (\text{Real} \rightarrow \text{Real}) \rightarrow \text{Real}$

Consider the self-difference $\zeta$ for $C$

$$\zeta \in [\text{real} \rightarrow \text{real}] \times ([\text{real} \rightarrow \text{real}] \rightarrow |\text{real}|) \rightarrow |\text{real}|$$

$$\zeta = \lambda (\varphi, \psi). \psi(\varphi(c), \psi(c, 0)).$$

What is the impact of replacing $e_{\sin}$ with $e_{id}$ in context $C$?

$$\zeta([e_{id}], \lambda (r, \varepsilon). \varepsilon + |\sin r - r|)$$
Our analysis is compositional

We have taken the context $C$ into account, but once and for all

The map $\zeta$ can be computed without any reference to $e_{id}$ and $e_{\sin}$
Theoretical Results on Differential Logical Relations
# DLRs and Ordinary LRs

## Hereditary Null Differences

DLRs subsume ordinary LRs

\[
\begin{align*}
(\text{real})^0 &= \{0\} \\
(\sigma \times \tau)^0 &= (\sigma)^0 \times (\tau)^0 \\
(\sigma \to \tau)^0 &= \{\phi \in (\sigma \to \tau) \mid \forall x \in \llbracket \sigma \rrbracket. \forall \zeta \in (\sigma)^0. \phi(x, \zeta) \in (\tau)^0\}
\end{align*}
\]

## Proposition

Two programs \( \vdash e, f : \tau \) are logically related iff \( \exists \phi \in (\tau)^0. D_\tau(e, \phi, f) \)
DLRs and Metric LRs

DLRs subsumes **metric** LRs

**Hereditary Real-valued Differences**

We parametrize $|\tau|$ by a single real number $r$

$(\text{real})^r = \{r\}$

$(\sigma \times \tau)^r = (\sigma)^r \times (\tau)^r$

$(\sigma \rightarrow \tau)^r = \{\phi \in (\sigma \rightarrow \tau) \mid \forall x \in [\sigma]. \forall \zeta \in (\sigma)^s. \phi(x, \zeta) \in (\tau)^r + s\}$

**Proposition**

For all programs $\vdash e, f : \tau$, $\delta^L(e, f) = r$ iff $\exists \phi \in (\tau)^r \cdot D_\tau(e, \phi, f)$
Since the calculus is strongly normalizing we expect differences to be hereditary finite.

<table>
<thead>
<tr>
<th>Hereditary Finite Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>((\text{real})^{&lt;\infty} = \mathbb{R}_{\geq 0})</td>
</tr>
<tr>
<td>((\sigma \times \tau)^{&lt;\infty} = (\sigma)^{&lt;\infty} \times (\tau)^{&lt;\infty})</td>
</tr>
<tr>
<td>((\sigma \rightarrow \tau)^{&lt;\infty} = { \phi \in (\sigma \rightarrow \tau) \mid \forall x \in [\sigma]. \forall \zeta \in (\sigma)^{&lt;\infty}. \phi(x, \zeta) \in (\tau)^{&lt;\infty}} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fundamental Lemma, II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assume all real-valued operators (F) to denote a weakly bounded function (F : \mathbb{R}^n \rightarrow \mathbb{R}) (i.e. (F(B)) bounded whenever (B \subseteq \mathbb{R}^n) is). Then, for any program (\vdash e : \tau) there exists (\phi \in (\tau)^{&lt;\infty}). (D_\tau(e, \phi, e))</td>
</tr>
</tbody>
</table>
**Categorical Foundation**

Metric LRs have a categorical foundation in the symmetric monoidal closed category of pseudometric spaces and Lipschitz-continuous maps.

Also DLRs have a categorical foundation in the category of generalized metric domains.

**Definition**

Given a quantale \((V, \leq, \oplus, 0)\), a generalised metric domain on \(V\) is a pair \((X, \mathcal{D}_X)\), where \(X\) is a set and \(\mathcal{D}_X \subseteq X \times V \times X\) satisfies:

\[
\mathcal{D}_X(x, 0, y) \implies x = y
\]

\[
\mathcal{D}_X(x, \varphi, y) \implies \mathcal{D}(y, \varphi, x)
\]

\[
\mathcal{D}_X(x, \varphi, y) \land \mathcal{D}_X(y, \chi, y) \land \mathcal{D}(y, \xi, z) \implies \mathcal{D}_X(x, \varphi \oplus \chi \oplus \xi, z)
\]
We have non-null self distance $D_X(x, \varphi, x) \neq 0 \iff \varphi = 0$

**Theorem**

GMDs form a **cartesian closed category**

Arrows are defined following the defining of DLRs
Conclusion
Conclusion

We have introduced DLRs and their basic properties.

We show the strengths of DLRs studying context-aware approximate program transformations.

But DLRs are interesting also from a purely theoretical perspective.
Differential Program Semantics

DLRs are the first step towards differential program semantics

ERC Consolidator Grant DIAPASoN

Focus on program difference rather than program identity

- Ugo Dal Lago
- Simone Martini
- Davide Sangiorgi
- Aurore Alcolei
- Guillaume Geoffroy
- Paolo Pistone
- Melissa Antonelli
- Gabriele Vannoni
Several future works

• Full recursion (step-indexing?)
• Connections with incremental computing?
• Differential bisimulation? Differential game semantics?
• Effects
Future Work: Probabilistic Approximate Program Transformation

Probabilistic loop perforation: compute $F(x_i)$ only for $s \leq n$ randomly sampled inputs

Probabilistic substitution

$$\lambda x.\sin x \xrightarrow{T} (\lambda x.\sin x) \oplus (\lambda x.x)$$
Questions
The Category of Generalized Metric Domains

**Objects** Triples \((X, \mathcal{D}_X, V)\)

**Arrows** \((f, \Gamma) : (X, \mathcal{D}_X, V_X) \to (Y, \mathcal{D}_Y, V_Y)\)

\[
f : X \to Y \quad \Gamma : X \times V_X \to V_Y
\]

such that

\[
\mathcal{D}_X(x, \varphi, x') \implies \begin{cases} 
\mathcal{D}_Y(f(x), \Gamma(x, \varphi), f(x')) \\
\mathcal{D}_Y(f(x), \Gamma(x', \varphi), f(x'))
\end{cases}
\]