Gradual Parametricity, Revisited

Éric Tanter

Inria Paris & University of Chile

joint work with
Matías Toro & Elizabeth Labrada

1

Gradual Typing 101

precision

$\mathsf{Int} \to \mathsf{Int} \sqsubseteq \mathsf{Int} \to ? \sqsubseteq ? \to ? \sqsubseteq ?$

some gradual types convey more information

consistency

best-effort static checking
backed by dynamic checking

$\mathtt{Int} \to ? \sim ? \to \mathtt{Bool}$

$\mathtt{Int} \to ? \not\sim \mathtt{Bool} \to ?$

type safety (+ errors)

embedding of “dynamic language”

conservative extension of “static language”

gradual guarantees SGG & DGG

2

Challenging Gradual Typing

subtyping

parametric polymorphism

unions

recursive types

ownership

refinements

typestates

effects

security typing

session types

3

Parametric Polymorphism

System F
explicit

```
f : ∀X. X → X
f = ΛX.λx:X.x
f [Int]  10
f [Bool] true
```

add type binders

infer type instantiations

Haskell
implicit

```
f : X → X
f = λx.x
f 10
f true
```

implicit polymorphism induces a notion of subtyping

$\forall X.X \to X <: \mathtt{Int} \to \mathtt{Int}$ $\qquad$ $\mathtt{Int} \to \mathtt{Int} <: \forall X.\mathtt{Int} \to \mathtt{Int}$

add [Int] $\qquad$ add ΛX

4

Parametricity 101

- Flavors of parametric polymorphism
  - **Genericity**: type safety with generic definitions
  - **Parametricity**: strong reasoning principle ("theorems for free")

	G	P
$f : \forall X.X \to X$		
$f\ [\texttt{Int}]\ n$	n'	n
$g : \forall XY.X \to Y \to X$		
$g\ [\texttt{Int}]\ [\texttt{Int}]\ n_1\ n_2$	n'	n_1

Gradual Parametricity

```
g : (∀X.X→X) → Int
g = λh:(∀X.X→X).h [Int] 10
```

```
f : ?
g f
```

```
f = ΛX.λx:X.x
f = ΛX.λx:?.x
f = ΛX.λx:?.x+1
```

- Gradual parametricity is challenging
  - early work on PBC [POPL'11] without proof of parametricity
  - several recent developments:
    λB [ICFP'17], System F_G [ICFP'17], Xie et al [ESOP'18]
  - arguable decisions, problems, conjectures (dynamic GG?)
  - free theorems not fully understood

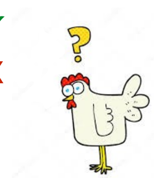
design space is subtle and complex

Gradual Parametricity

λB and F_G use ad hoc static relations

- "explicit" polymorphism with (different) flavors of implicitness

		λB	System F_G
$\forall X.X \to X$	$\texttt{Int} \to \texttt{Int}$	✓	X
	$? \to ?$	✓	✓
	$\texttt{Int} \to \texttt{Bool}$	✓	X
$\forall X.X \to \texttt{Int}$	$\forall X.X \to ?$	✓	✓
	$\forall X.? \to \texttt{Int}$	✓	X



Gradual Implicit Polymorphism

[Xie et al. ESOP'18]

- same observations wrt statics of λB and F_G
- separate consistency from subtyping
- clean definition of consistent subtyping
- focus on the statics, uses λB for runtime

Gradual Parametricity @ Runtime

- λB as the target of choice for runtime semantics
  - violates scoping & type instantiation

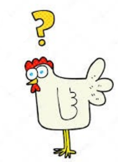
9

Gradual Parametricity @ Runtime

- λB as the target of choice for runtime semantics
  - violates **scoping** & type instantiation

```
let f : ? = ΛX.λx:X.x
```

	λB	System F_G
f [Int] 1	1^*	1^*
(f [Int] 1) + 1	**error**	**error**



pending cast with out-of-scope variable
(λB) (1 : Int => α : α => ?)
(System F_G) (1 : X => ?)

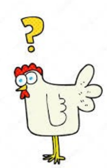
10

Gradual Parametricity @ Runtime

- λB as the target of choice for runtime semantics
  - violates scoping & **type instantiation**

```
let f : ? = ΛX.λx:X.x
```

	λB	System F_G
f [Bool] true	true	true
f [Int] true	true	true



instantiations on ? are lost!

11

Instantiations matter!

gradual types **soundly augment expressiveness**

	t = λx.x		**t = λx.(x [Int])**
STLC	Int → Int Bool → Bool …	F	(∀X.X→X) → Int→Int (∀XY.X→Y→X) → ∀Y.Int→Y→Int …
F	∀X.X → X	Fω	∀P, (∀X.P X) → P Int
GTLC	? → ?	GF	(∀X.?) → ?

lack of precision backed by **runtime enforcement**

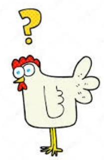
(t 3) + 1 ⇒ **4**	id:∀X.X→X	(t id) 1	⇒ **1**
(t 3) 1 ⇒ **error**		(t id) true	⇒ **error**

12

Gradual Free Theorems

- λB [ICFP'17] includes some free theorems
  - about fully-static signatures
- What about imprecise type signatures?
  - claims only
  - ∀X.X→?
    constant or fail [POPL'11], or unusable result [STOP'09,ICFP'17]
  - ∀X.?→X
    always fail [ICFP'17]

id?



13

Open Challenges

- Proper semantics for gradual explicit polymorphism?
- λB (/ F_C) adequate as an internal cast language?
- Parametricity &/vs dynamic gradual guarantee?
- Gradual free theorems?

14

This Work

- Proper semantics for gradual explicit polymorphism?
  GSF: derived from F using AGT (++)
- λB (/ F_C) adequate as an internal cast language?
  no: type instantiations matter (GSFε)
- Parametricity &/vs dynamic gradual guarantee?
  incompatible! (weaker property holds)
- Gradual free theorems?
  disprove claims, prove "cheap" theorems

15

GSF: Gradual System F

16

Gradual Parametricity, Revisited

- Exploit methodology to gradualize languages: AGT
  - effective: refinements, effects, unions, security typing, etc.
- Natural definition of gradual types
  - static semantics follow via Galois connection (no tweaking!)
- Dynamic semantics
  - evidence-based reduction semantics
  - strengthened for parametricity enforcement

Concretization for GSF

(syntactic) meaning of gradual types

$$\gamma(\texttt{Int}) = \{\, \texttt{Int} \,\}$$
$$\gamma(G_1 \to G_2) = \{\, T_1 \to T_2 \mid T_1 \in \gamma(G_1), T_2 \in \gamma(G_2) \,\}$$
$$\gamma(?) = \{\, T \mid T \in \textsc{Type} \,\}$$
$$\gamma(X) = \{\, X \,\}$$
$$G) = \{\, \forall X.T \mid T \in \gamma(G) \,\}$$

induces precision…

$$G_1 \sqsubseteq G_2 \quad \triangleq \quad \gamma(G_1) \subseteq \gamma(G_2)$$

… and consistency

$$G_1 \sim G_2 \qquad \begin{matrix} \sqcup\!\shortmid & \sqcup\!\shortmid \end{matrix} \qquad T_1 = T_2$$

$$\forall X.X \to X \;\sqsubseteq\; \forall X.X \to ? \;\sqsubseteq\; \forall X.? \to ? \;\sqsubseteq\; ?$$
$$\forall X.? \to X$$

GSF Statics

		λB	System F_G	GSF
$\forall X.X \to X$	$\texttt{Int} \to \texttt{Int}$	✓	X	X
	$? \to ?$	✓	✓	X
	$\texttt{Int} \to \texttt{Bool}$	✓	X	X
$\forall X.X \to \texttt{Int}$	$\forall X.X \to ?$	✓	✓	✓
	$\forall X.? \to \texttt{Int}$	✓	X	✓

conservative extension of System F
explicit polymorphism
"natural" precision & consistency

Evidence-based Dynamics

- keep track of witnesses of consistent judgments
- consistent transitivity: combine evidence or fail

$(\lambda x : ?.x + 1)$ false

evidence-augmented term

$$(\varepsilon_{? \to \mathsf{Int}}(\lambda x : ?.(\varepsilon_1 x :: \mathsf{Int}) + (\varepsilon_{\mathsf{Int}} 1 :: \mathsf{Int})) :: ? \to \mathsf{Int})\ (\varepsilon_2(\varepsilon_{\mathsf{Bool}} \mathsf{false} :: \mathsf{Bool}) :: ?)$$

justifies ? ~ Int

consistent transitivity

justifies Bool ~ ?

$$\Xi \triangleright \varepsilon_2(\varepsilon_1 u :: G_1) :: G_2 \quad \longrightarrow \quad \begin{cases} \Xi \triangleright (\varepsilon_1 \circ \varepsilon_2) u :: G_2 & \\ \textbf{error} & \text{if not defined} \end{cases}$$

GSF Dynamics

- Refinements for GSF
  - type names in evidence: path of bindings
  - strengthen CT to enforce parametricity

$\langle \texttt{Int}, \alpha^{\texttt{Int}} \rangle$ seal

$\langle \alpha^{\texttt{Int}}, \texttt{Int} \rangle$ unseal

$$\Xi \triangleright (\varepsilon \Lambda X.t :: \forall X.G)\ [G'] \longrightarrow \Xi' \triangleright \varepsilon_{unsl}(\varepsilon[\hat{\alpha}]t[\hat{\alpha}/X] :: G[\alpha/X]) :: G[G'/X]$$

unseal on the outside

GSF Dynamics

```
let f : ? = ΛX.λx:X.x
```

	λB	System F_G	GSF
`f [Int] 1`	1	1	1
`(f [Int] 1) + 1`	error	error	2
`f [Bool] true`	true	true	true
`f [Int] true`	true	true	error

proper handling of type instantiations wrt ?

GSF Properties

- Type safe, conservative extension of System F
- Well-typed terms are relationally-parametric [Ahmed+17]
- Dynamic gradual guarantee *cannot* be satisfied together with parametricity (pick one)

Parametricity™ is incompatible with DGG™

```
f : ∀X.X→X =          f' : ∀X.?→X =
  ΛX.λx:X.x::X    ⊑     ΛX.λx:?.x::X
```

must necessarily fail when applied

proof independent of representation of evidence and of the interpretation of unknown type

GSF: System F imprecise embedding

A System F term ascribed to an imprecise GSF type hereditarily terminates

$$\text{If} \vdash t : T \text{ and } T \sqsubseteq G, \text{ then} \vdash (t :: G) \rightsquigarrow t' : G \text{ and} \models t' : T \sqsubseteq G.$$

$$\Sigma \triangleright t \longmapsto^* \Sigma' \triangleright v \wedge v \in \mathcal{N}_\rho^{\Sigma'} [\![T \sqsubseteq G]\!]$$

new property, weaker than DGG
"the gradual medium is harmless"

Gradual Free Claims Disproved

imprecise termination has interesting consequences

- ∀X.X→?
  constant or fail [POPL'11], or unusable result [STOP'09,ICFP'17]
- ∀X.?→X
  always fail [ICFP'17]

id :: ∀X.X→?
id :: ∀X.?→X
are both well-behaved identity functions!

"Cheap Theorems"

when looking at the types is not interesting enough…

(v: ∀X.?→X) ∧ (v = ΛX.λx:?.t) ⇒ always fails

independent of body

Conclusions

State-of-the-art

	language	polym.	static relations	parametricity	static GG	dynamic GG
PBC Ahmed et al. POPL'11	cast calculus	explicit*	"ad-hoc" compatibility	conjecture	-	-
λB Ahmed et al. ICFP'17	cast calculus	explicit*	"ad-hoc" compatibility	proven	(future)	(future)
F_G / F_C Igarashi et al. ICFP'17	source language	explicit*	"ad-hoc" precision & consistency	conjecture (F_C)	proven*	conjecture
Xie et al. ESOP'18	source language	implicit	consistent subtyping	inherited (λ_B)	proven	-

Contribution

new criteria violated by all existing work

	language	polym.	static relations	parametricity	static GG	dynamic GG	faithful instantiation
PBC Ahmed et al. POPL'11	cast calculus	explicit*	"ad-hoc" compatibility	conjecture	-	-	X
λB Ahmed et al. ICFP'17	cast calculus	explicit*	"ad-hoc" compatibility	proven	(future)	(future)	X
F_G / F_C Igarashi et al. ICFP'17	source language	explicit*	"ad-hoc" precision & consistency	conjecture (F_C)	proven*	conjecture	X
Xie et al. ESOP'18	source language	implicit	consistent subtyping	inherited (λ_B)	proven	-	X
GSF	**source language**	**explicit**	**precision & consistency**	**proven**	**proven**	**proven impossible**	✓

sensible, systematic design (derived with AGT)

can swap

Contributions beyond GSF

1. λB (and System F_C) are **not** appropriate internal languages of gradual parametricity
2. parametricity™ is **incompatible** with DGG™
3. novel: imprecise embedding of F, disprove claims

so is noninterference! (language with references) [TOPLAS]

Perspectives

- Cast calculus for GSF
  - prove equivalence with evidence-based semantics
- Add implicit polymorphism
  - combine with [Xie et al.]
- Novel form of parametricity to reconcile with DGG?