Algebraic subtyping for algebraic effects and handlers

Axel Faes
student : undergraduate
ACM Student Member: 2461936
Department of Computer Science
KU Leuven
axel.faes@student.kuleuven.be

Amr Hany Saleh
daily advisor
Department of Computer Science
KU Leuven
amrhanyshehata.saleh@kuleuven.be

Tom Schrijvers
promotor
Department of Computer Science
KU Leuven
tom.schrijvers@kuleuven.be

Abstract

Keywords algebraic effect handler, algebraic subtyping, effects, optimised compilation

1 Introduction

2 Background (Eff)

The type-&-effect system that is used in Eff is based on subtyping and dirty types [1].

2.1 Types and terms

Terms Figure 1 shows the two types of terms in Eff. There are values v and computations c. Computations are terms that can contain effects. Effects are denoted as operations Op which can be called.

Types Figure 2 shows the types of Eff. There are two main sorts of types. There are (pure) types A, B and dirty types $\underline{C}, \underline{D}$. A dirty type is a pure type A tagged with a finite set of operations Δ, which we call dirt, that can be called. This finite set Δ is an over-approximation of the operations that are actually called. The type $\underline{C} \Rrightarrow \underline{D}$ is used for handlers because a handler takes an input computation $\underline{C}$, handles the effects in this computation and outputs computation $\underline{D}$ as the result.

$$
\begin{array}{llll}
\text{value } v & ::= & x & \text{variable} \\
 & | & \mathsf{k} & \text{constant} \\
 & | & \mathtt{fun}\ x \mapsto c & \text{function} \\
 & | & \{ & \text{handler} \\
 & & \quad \mathtt{return}\ x \mapsto c_r, & \quad \text{return case} \\
 & & \quad [\mathtt{Op}\ y\ k \mapsto c_{\mathtt{Op}}]_{\mathtt{Op} \in O} & \quad \text{operation cases} \\
 & & \} & \\
\text{comp } c & ::= & v_1\, v_2 & \text{application} \\
 & | & \mathtt{let\ rec}\ f\ x = c_1\ \mathtt{in}\ c_2 & \text{rec definition} \\
 & | & \mathtt{return}\ v & \text{returned val} \\
 & | & \mathtt{Op}\ v & \text{operation call} \\
 & | & \mathtt{do}\ x \leftarrow c_1\ ;\ c_2 & \text{sequencing} \\
 & | & \mathtt{handle}\ c\ \mathtt{with}\ v & \text{handling}
\end{array}
$$

Figure 1. Terms of Eff

$$
\begin{array}{rcll}
\text{(pure) type } A, B & ::= & \mathtt{bool} \mid \mathtt{int} & \text{basic types} \\
 & | & A \to \underline{C} & \text{function type} \\
 & | & \underline{C} \Rrightarrow \underline{D} & \text{handler type} \\
\text{dirty type } \underline{C}, \underline{D} & ::= & A \,!\, \Delta & \\
\text{dirt } \Delta & ::= & \{\mathtt{Op}_1, \ldots, \mathtt{Op}_n\} &
\end{array}
$$

Figure 2. Types of Eff

Subtyping

$$
\textsc{Sub-bool}\ \frac{}{\mathtt{bool} \leqslant \mathtt{bool}}
\qquad
\textsc{Sub-int}\ \frac{}{\mathtt{int} \leqslant \mathtt{int}}
\qquad
\textsc{Sub-}{\to}\ \frac{A' \leqslant A \qquad \underline{C} \leqslant \underline{C}'}{A \to \underline{C} \leqslant A' \to \underline{C}'}
$$

$$
\textsc{Sub-}{\Rrightarrow}\ \frac{\underline{C}' \leqslant \underline{C} \qquad \underline{D} \leqslant \underline{D}'}{\underline{C} \Rrightarrow \underline{D} \leqslant \underline{C}' \Rrightarrow \underline{D}'}
\qquad
\textsc{Sub-!}\ \frac{A \leqslant A' \qquad \Delta \subseteq \Delta'}{A \,!\, \Delta \leqslant A' \,!\, \Delta'}
$$

Figure 3. Subtyping for pure and dirty types of Eff

2.2 Type System

2.2.1 Subtyping

The dirty type $A \,!\, \Delta$ is assigned to a computation returning values of type A and potentially calling operations from the set Δ. This set Δ is always an over-approximation of the actually called operations, and may safely be increased, inducing a natural subtyping judgement $A \,!\, \Delta \leq A \,!\, \Delta'$ on dirty types. As dirty types can occur inside pure types, we also get a derived subtyping judgement on pure types. Both judgements are defined in Figure 3. Observe that, as usual, subtyping is contravariant in the argument types of functions and handlers, and covariant in their return types.

2.2.2 Typing rules

Figure 4 defines the typing judgements for values and computations with respect to a standard typing context Γ.

Values The rules for subtyping, variables, and functions are entirely standard. For constants we assume a signature Σ that assigns a type A to each constant k, which we write as $(\mathsf{k} : A) \in \Sigma$.

A handler expression has type $A \,!\, \Delta \cup O \Rightarrow B \,!\, \Delta$ iff all branches (both the operation cases and the return case) have dirty type $B \,!\, \Delta$ and the operation cases cover the set of operations O. Note that the intersection $\Delta \cap O$ is not necessarily empty. The handler deals with the operations O, but in the process may re-issue some of them (i.e., $\Delta \cap O$).

When typing operation cases, the given signature for the operation $(\mathsf{Op} : A_{\mathsf{Op}} \to B_{\mathsf{Op}}) \in \Sigma$ determines the type A_{Op} of the parameter x and the domain B_{Op} of the continuation k. As our handlers are deep, the codomain of k should be the same as the type $B \,!\, \Delta$ of the cases.

Computations With the following exceptions, the typing judgement $\Gamma \vdash c : \underline{C}$ has a straightforward definition. The `return` construct renders a value v as a pure computation, i.e., with empty dirt. An operation invocation $\mathsf{Op}\,v$ is typed according to the operation's signature, with the operation itself as its only operation. Finally, rule With shows that a handler with type $\underline{C} \Rightarrow \underline{D}$ transforms a computation with type $\underline{C}$ into a computation with type $\underline{D}$.

3 Core language

$\sqcup$ is for outputs, $\sqcap$ is for inputs.

3.1 Types and terms

3.2 Type system

3.2.1 Subtyping

3.2.2 Typing rules

4 Elaboration

5 Proofs

6 Conclusion

Acknowledgments

I would like to thank Amr Hany Saleh for his continuous guidance and help.

References

[1] Andrej Bauer and Matija Pretnar. 2014. An Effect System for Algebraic Effects and Handlers. *Logical Methods in Computer Science* 10, 4 (2014). https://doi.org/10.2168/LMCS-10(4:9)2014

$$\text{typing contexts } \Gamma \;::=\; \epsilon \;\mid\; \Gamma, x : A$$

Expressions

$$\textsc{SubVal}\ \frac{\Gamma \vdash v : A \qquad A \leqslant A'}{\Gamma \vdash v : A'} \qquad \textsc{Var}\ \frac{(x : A) \in \Gamma}{\Gamma \vdash x : A} \qquad \textsc{Const}\ \frac{(\mathsf{k} : A) \in \Sigma}{\Gamma \vdash \mathsf{k} : A}$$

$$\textsc{Fun}\ \frac{\Gamma, x : A \vdash c : \underline{C}}{\Gamma \vdash \mathsf{fun}\ x \mapsto c : A \to \underline{C}}$$

$$\textsc{Hand}\ \frac{\Gamma, x : A \vdash c_r : B \,!\, \Delta \qquad \left[(\mathsf{Op} : A_{\mathsf{Op}} \to B_{\mathsf{Op}}) \in \Sigma \quad \Gamma, x : A_{\mathsf{Op}}, k : B_{\mathsf{Op}} \to B \,!\, \Delta \vdash c_{\mathsf{Op}} : B \,!\, \Delta\right]_{\mathsf{Op} \in O}}{\Gamma \vdash \{\mathtt{return}\ x \mapsto c_r, [\mathsf{Op}\, y\, k \mapsto c_{\mathsf{Op}}]_{\mathsf{Op} \in O}\} : A \,!\, \Delta \cup O \Rightarrow B \,!\, \Delta}$$

Computations

$$\textsc{SubComp}\ \frac{\Gamma \vdash c : \underline{C} \qquad \underline{C} \leqslant \underline{C}'}{\Gamma \vdash c : \underline{C}'} \qquad \textsc{App}\ \frac{\Gamma \vdash v_1 : A \to \underline{C} \qquad \Gamma \vdash v_2 : A}{\Gamma \vdash v_1\, v_2 : \underline{C}}$$

$$\textsc{LetRec}\ \frac{\Gamma, f : A \to \underline{C}, x : A \vdash c_1 : \underline{C} \qquad \Gamma, f : A \to \underline{C} \vdash c_2 : \underline{D}}{\Gamma \vdash \mathtt{let\ rec}\ f\ x = c_1\ \mathtt{in}\ c_2 : \underline{D}}$$

$$\textsc{Ret}\ \frac{\Gamma \vdash v : A}{\Gamma \vdash \mathsf{return}\ v : A \,!\, \emptyset} \qquad \textsc{Op}\ \frac{(\mathsf{Op} : A \to B) \in \Sigma \qquad \Gamma \vdash v : A}{\Gamma \vdash \mathsf{Op}\, v : B \,!\, \{\mathsf{Op}\}}$$

$$\textsc{Do}\ \frac{\Gamma \vdash c_1 : A \,!\, \Delta \qquad \Gamma, x : A \vdash c_2 : B \,!\, \Delta}{\Gamma \vdash \mathsf{do}\ x \leftarrow c_1 \,;\, c_2 : B \,!\, \Delta}$$

$$\textsc{With}\ \frac{\Gamma \vdash v : \underline{C} \Rightarrow \underline{D} \qquad \Gamma \vdash c : \underline{C}}{\Gamma \vdash \mathsf{handle}\ c\ \mathtt{with}\ v : \underline{D}}$$

Figure 4. Typing of Eff

value v	$::=$	x	variable
	$\mid$	k	constant
	$\mid$	$\lambda x.c$	function
	$\mid$	$\Lambda\alpha.c$	type abstraction
	$\mid$	$\{$	handler
		$\quad\mathtt{return}\ x \mapsto c_r,$	return case
		$\quad[\mathtt{Op}\ y\ k \mapsto c_{\mathtt{Op}}]_{\mathtt{Op}\in O}$	operation cases
		$\}$	
comp c	$::=$	$v_1\ v_2$	application
	$\mid$	$\mathtt{if}\ e\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2$	conditional
	$\mid$	$v\ A$	type application
	$\mid$	$\mathtt{let\ rec}\ f\ x = c_1\ \mathtt{in}\ c_2$	rec definition
	$\mid$	$\mathtt{return}\ v$	returned val
	$\mid$	$\mathtt{Op}\ v$	operation call
	$\mid$	$\mathtt{do}\ x \leftarrow c_1\ ;\ c_2$	sequencing
	$\mid$	$\mathtt{handle}\ c\ \mathtt{with}\ v$	handling

Figure 5. Terms of the core language

(pure) type A, B	$::=$	$\mathtt{bool} \mid \mathtt{int}$	basic types
	$\mid$	$A \to \underline{C}$	function type
	$\mid$	$\underline{C} \Rightarrow \underline{D}$	handler type
	$\mid$	α	type variable
	$\mid$	$\forall\alpha.\underline{C}$	polytype
	$\mid$	$\top$	top
	$\mid$	$\bot$	bottom
	$\mid$	$A \sqcap B$	intersection
	$\mid$	$A \sqcup B$	union
dirty type $\underline{C}, \underline{D}$	$::=$	$A \mathbin{!} \Delta$	
dirt Δ	$::=$	$\{\mathrm{R}\}$	
R	$::=$	$\mathtt{Op}\ ;\ R$	row
	$\mid$	δ	row variable
	$\mid$	$.$	closed row
	$\mid$	$R_1 \sqcap R_2$	intersection
	$\mid$	$R_1 \sqcup R_2$	union

Figure 6. Types of the core language

Subtyping

Sub-bool
$$\frac{}{\mathtt{bool} \leqslant \mathtt{bool}}$$

Sub-int
$$\frac{}{\mathtt{int} \leqslant \mathtt{int}}$$

Sub-$\to$
$$\frac{A' \leqslant A \qquad \underline{C} \leqslant \underline{C}'}{A \to \underline{C} \leqslant A' \to \underline{C}'}$$

Sub-$\Rightarrow$
$$\frac{\underline{C}' \leqslant \underline{C} \qquad \underline{D} \leqslant \underline{D}'}{\underline{C} \Rightarrow \underline{D} \leqslant \underline{C}' \Rightarrow \underline{D}'}$$

Sub-!
$$\frac{A \leqslant A' \qquad \Delta \subseteq \Delta'}{A \mathbin{!} \Delta \leqslant A' \mathbin{!} \Delta'}$$

Sub-Inter-G
$$\frac{A \leqslant B \qquad A \leqslant B'}{A \leqslant B \sqcap B'}$$

Sub-Inter-LB
$$\frac{}{A_1 \sqcap A_2 \leqslant A_i}$$

Sub-Union-L
$$\frac{B \leqslant A \qquad B' \leqslant A}{B \sqcup B' \leqslant A}$$

Sub-Inter-UB
$$\frac{}{A_i \leqslant A_1 \sqcup A_2}$$

Sub-Dist-Func-Inter
$$\frac{}{(A \to \underline{C}) \sqcap (A \to \underline{D}) \leqslant A \to (\underline{C} \sqcap \underline{D})}$$

Sub-Dist-Func-Union
$$\frac{}{(A \to \underline{C}) \sqcap (B \to \underline{C}) \leqslant (A \sqcup B) \to \underline{C}}$$

Sub-Dist-Hand-Inter
$$\frac{}{(A \Rightarrow \underline{C}) \sqcap (A \Rightarrow \underline{D}) \leqslant A \Rightarrow (\underline{C} \sqcap \underline{D})}$$

Sub-Dist-Hand-Union
$$\frac{}{(A \Rightarrow \underline{C}) \sqcap (B \Rightarrow \underline{C}) \leqslant (A \sqcup B) \Rightarrow \underline{C}}$$

Sub-Inter-G-!
$$\frac{\Delta \subseteq \Delta' \qquad \Delta \subseteq \Delta''}{\Delta \subseteq (\Delta' \sqcap \Delta'')}$$

Sub-Inter-LB-!
$$\frac{}{(\Delta_1 \sqcap \Delta_2) \subseteq \Delta_i}$$

Sub-Union-L-!
$$\frac{\Delta' \subseteq \Delta \qquad \Delta'' \subseteq \Delta}{(\Delta' \sqcup \Delta'') \subseteq \Delta}$$

Sub-Union-UB-!
$$\frac{}{\Delta_i \subseteq (\Delta_1 \sqcup \Delta_2)}$$

Sub-Top
$$\frac{}{A \leqslant \top}$$

Sub-Bottom
$$\frac{}{\bot \leqslant A}$$

Figure 7. Subtyping for pure and dirty types of the core language

$$\text{typing contexts } \Gamma \ ::= \ \epsilon \ | \ \Gamma, x : A$$

Expressions

$$\textsc{Val}\ \frac{\Gamma \vdash v : A \qquad A \leqslant B}{\Gamma \vdash v : B} \qquad \textsc{Var}\ \frac{(x : A) \in \Gamma}{\Gamma \vdash x : A} \qquad \textsc{Const}\ \frac{(\mathsf{k} : A) \in \Sigma}{\Gamma \vdash \mathsf{k} : A}$$

$$\textsc{Fun}\ \frac{\Gamma, x : A \vdash c : \underline{C}}{\Gamma \vdash \lambda x.c : A \to \underline{C}} \qquad \textsc{Type Abstraction}\ \frac{\Gamma, \alpha \vdash c : \underline{C}}{\Gamma \vdash \Lambda\alpha.c : \forall\alpha.\underline{C}}$$

$$\textsc{Hand}\ \frac{\begin{array}{c}\underline{C} = A \,!\, \{Op_i \,;\, R\} \\ \underline{D} = B \,!\, \{Op_i \,;\, R\} \qquad (\mathsf{Op}_i : A_{\mathsf{Op}} \to B_{\mathsf{Op}}) \in \Sigma \\ h = \{\mathtt{return}\ x \mapsto c_r, [\mathsf{Op}\, y\, k \mapsto c_{\mathsf{Op}}]_{\mathsf{Op} \in O}\} \\ \Gamma, x : A_{\mathsf{Op}} \vdash c_r : \underline{D} \\ \Gamma, y : A_{\mathsf{Op}}, k : B_{\mathsf{Op}} \to \underline{D} \vdash c_{op} : \underline{D}\end{array}}{\Gamma \vdash h : \underline{C} \Rightarrow \underline{D}}$$

Computations

$$\textsc{Comp}\ \frac{\Gamma \vdash c : \underline{C} \qquad \underline{C} \leqslant \underline{D}}{\Gamma \vdash c : \underline{D}} \qquad \textsc{App}\ \frac{\Gamma \vdash v_1 : A \to \underline{C} \qquad \Gamma \vdash v_2 : A}{\Gamma \vdash v_1\, v_2 : \underline{C}}$$

$$\textsc{Cond}\ \frac{\Gamma \vdash v : A \qquad \Gamma \vdash c_1 : \underline{C} \qquad \Gamma \vdash c_2 : \underline{D}}{\Gamma \vdash \mathtt{if}\ v\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 : (\underline{C} \sqcup \underline{D})}$$

$$\textsc{Type App}\ \frac{\Gamma \vdash v : \forall\alpha.\underline{C}}{\Gamma \vdash v\, A : \underline{C}[A/\alpha]}$$

$$\textsc{LetRec}\ \frac{\Gamma, f : A \to \underline{C}, x : A \vdash c_1 : \underline{C} \qquad \Gamma, f : A \to \underline{C} \vdash c_2 : \underline{D}}{\Gamma \vdash \mathtt{let\ rec}\ f\, x = c_1\ \mathtt{in}\ c_2 : \underline{D}}$$

$$\textsc{Ret}\ \frac{\Gamma \vdash v : A}{\Gamma \vdash \mathtt{return}\ v : A \,!\, \emptyset}$$

$$\textsc{Op}\ \frac{(\mathsf{Op} : A \to B) \in \Sigma \qquad \Gamma \vdash v : A \qquad \underline{C} : B \,!\, \{\mathsf{Op} \,;\, R\}}{\Gamma \vdash \mathsf{Op}\, v : \underline{C}}$$

$$\textsc{Do}\ \frac{\Gamma \vdash c_1 : A \,!\, \Delta \qquad \Gamma, x : A \vdash c_2 : B \,!\, \Delta}{\Gamma \vdash \mathtt{do}\ x \leftarrow c_1 \,;\, c_2 : B \,!\, \Delta}$$

$$\textsc{With}\ \frac{\Gamma \vdash v : \underline{C} \Rightarrow \underline{D} \qquad \Gamma \vdash c : \underline{C}}{\Gamma \vdash \mathtt{handle}\ c\ \mathtt{with}\ v : \underline{D}}$$

Figure 8. Typing of the explicitly typed language