Nullable Type Inference

Michel Mauny, **Benoît Vaugon**
Ensta-ParisTech

Nullable Types

A **nullable type** $t?$ includes `NULL` and (unboxed) values of type t

$$\begin{array}{lcl} t & ::= & t_b \mid \alpha \mid \tau_1 \to \tau_2 \\ \tau & ::= & t \mid t? \end{array}$$

We provide a type inference algorithm:

- featuring Hindley-Milner polymorphism
- that statically guarantees that `NULL` cannot be used as a regular value
- whose soudness has been proved

Nullable types in practice

- Hack (Facebook)
- Swift (Apple)
- ...

```
func f (b : Bool, s : String) -> String? {
  if b { return s } else { return nil }
}
```

Type Inference

Swift:

```
var f = { (b : Bool, s : String) -> String? in
  if b { s } else { nil }
}
```

→ `Ok`

```
var f = { (b : Bool, s) -> String? in
  if b { s } else { nil }
}
```

→ `Error: Cannot convert type '(Bool, $T0) -> String?' to type '$T1'`

OCaml:

```
let f b (s : [ `String ]) =
  if b then s else `NULL
```

```
→ Error: This expression has type [> `NULL ]
         but an expression was expected of type [ `String ]
         The second variant type does not allow tag(s) `NULL
```

Replace Unification by Subtyping

The algorithm interleaves inference of subtyping constraints...

TApp

$$\frac{\texttt{let}\ \alpha\ \textit{fresh} \qquad \Phi, \Gamma \vdash e_1 : \alpha \to \tau \rhd \Phi' \qquad \Phi', \Gamma \vdash e_2 : \alpha \rhd \Phi''}{\Phi, \Gamma \vdash e_1\ e_2 : \tau \rhd \Phi''}$$

TNull

$$\frac{\texttt{let}\ \alpha\ \textit{fresh} \qquad \Phi \vdash \alpha? \leqslant \tau \rhd \Phi'}{\Phi, \Gamma \vdash \texttt{NULL} : \tau \rhd \Phi'}$$

TCase

$$\frac{\texttt{let}\ \alpha\ \textit{fresh} \qquad \Phi_0, \Gamma \vdash e_1 : \alpha? \rhd \Phi_1 \qquad \Phi_1, \Gamma \vdash e_2 : \tau \rhd \Phi_2 \qquad \Phi_2, \Gamma \oplus x : \alpha \vdash e_3 : \tau \rhd \Phi_3}{\Phi_0, \Gamma \vdash \texttt{case}\ e_1\ \texttt{of NULL} \to e_2 \mid x \to e_3 : \tau \rhd \Phi_3}$$

...(5 more rules)

...and their resolution

LeqArrow

$$\frac{\Phi \vdash \tau_1' \leqslant \tau_1 \rhd \Phi' \qquad \Phi' \vdash \tau_2 \leqslant \tau_2' \rhd \Phi''}{\Phi \vdash \tau_1 \to \tau_2 \leq \tau_1' \to \tau_2' \rhd \Phi''}$$

LeqBaseNull

$$\frac{}{\Phi \vdash t_b \leq t_b? \rhd \Phi}$$

LeqArrowNull

$$\frac{\Phi \vdash \tau_1' \leqslant \tau_1 \rhd \Phi' \qquad \Phi' \vdash \tau_2 \leqslant \tau_2' \rhd \Phi''}{\Phi \vdash \tau_1 \to \tau_2 \leq (\tau_1' \to \tau_2')? \rhd \Phi''}$$

GeqVarLeqTy

$$\frac{\text{when } \tau' \neq \alpha \text{ and } \tau' \leqslant \tau \notin \Phi \qquad \Phi, \alpha \leqslant \tau, \tau' \leqslant \tau \vdash \tau' \leq \alpha \rhd \Phi' \qquad \Phi' \vdash \tau' \leq \tau \rhd \Phi''}{\Phi, \alpha \leqslant \tau \vdash \tau' \leq \alpha \rhd \Phi''}$$

LeqVarEnd

$$\frac{\begin{array}{c}\text{when } \tau' \neq \alpha \qquad \text{when } (\forall \tau \mid \alpha \leqslant \tau \in \Phi \ \Rightarrow\ \tau \approx \tau' \in \Phi) \\ \text{when } (\forall \tau \mid \tau \leqslant \alpha \in \Phi \ \Rightarrow\ \tau \leqslant \tau' \in \Phi) \qquad \text{when } (\forall \tau \mid \alpha \approx \tau \in \Phi \ \Rightarrow\ \tau \approx \tau' \in \Phi)\end{array}}{\Phi \vdash \alpha \leq \tau' \rhd \Phi}$$

...(23 more rules)