TinyHM: Hindley-Milner type inference

Type systems and constraint solving

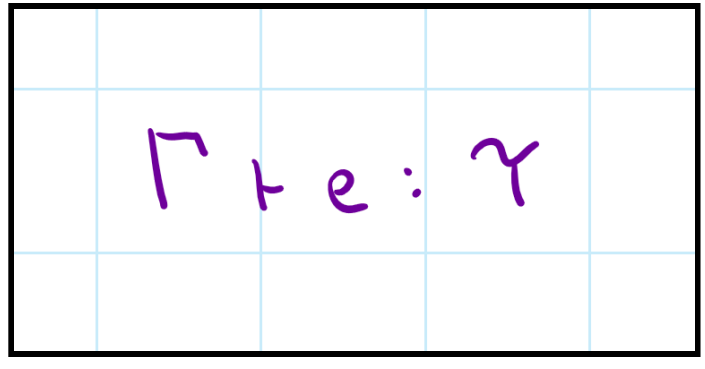
Tomas Petricek, Charles University

tomas@tomasp.net
@tomasp.net
https://tomasp.net
https://d3s.mff.cuni.cz/teaching/nprg077

Type systems

Typing rules

Given a typing context Γ, the expression e has a type τ



The problem in general

We know some of these,
want to figure out the rest

Charles University

Type systems

Type checking

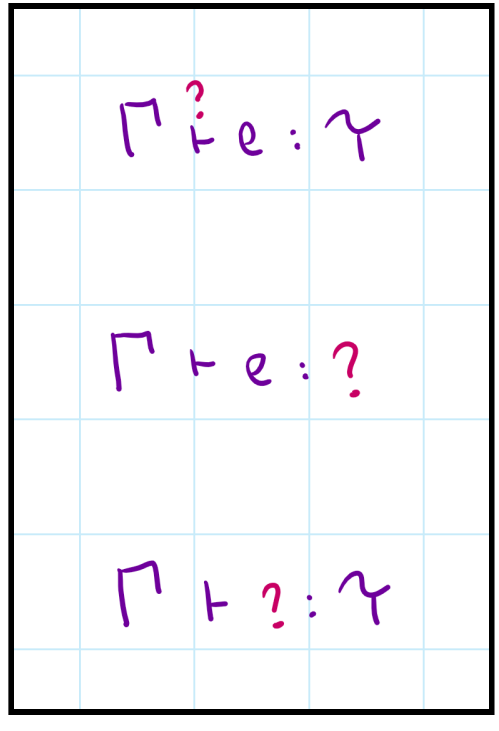
- Know it all. Check derivation exists!
- Easy for syntax-driven rules

Type inference

- Know expression. Figure out the type!
- Ideally most general (best) type

Program synthesis

- Not very common, but interesting idea!



Charles University

Type τ is the most general type for an expression e in context Γ if

- $\Gamma \vdash e : \tau$ and
- $\forall \sigma . \Gamma \vdash e : \sigma \rightarrow \sigma <: \tau$

Principal type (most general)

Best type of an expression

Any other type of the expression is a special case (subtype) of it

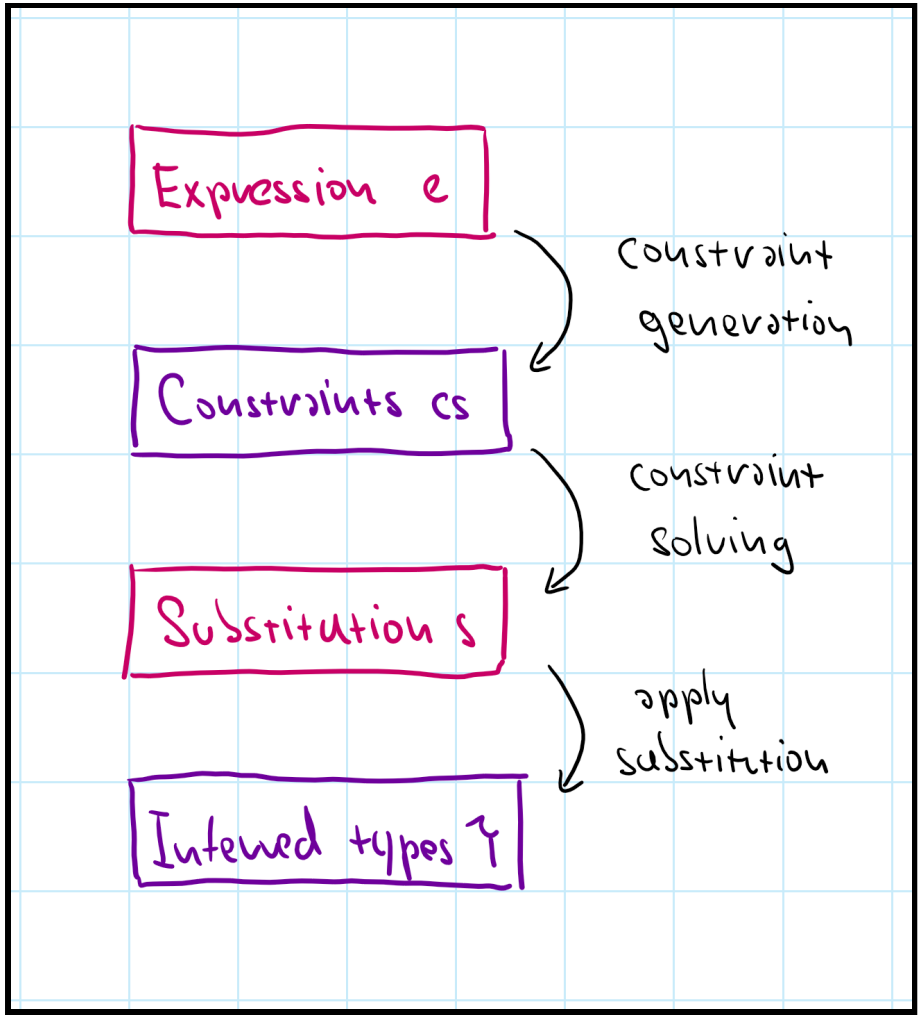
Charles University

Type inference

- ☑ How Hindely-Milner type inference works?
  Produces most general type (for ML)
- ☒ How Hindely-Milner type inference breaks?
  Nominal types with members, interfaces, etc.
- ? Alternative methods for type inference
  Bidirectional - combines checking and inference

Hindley-Milner

Constraint generation & solving



Two phase process

Generate constraints

Recursively over expression

Solve constraints

Recursively over constraint set

In the "Algorithm W", the two are combined. We separate them!

Charles University

```
(* Basic types with
   type variables *)
type Type =
  | TyNumber
  | TyVariable of string
  | TyFunction of Type * Type
  | TyList of Type

(* Constraint specifies
   that one type should be
   unified with another *)
type Constraint =
  Type * Type
```

What is a constraint?

A pair of types that should be unified

Easy or impossible

`int = int -> int`

`int list = int list`

Tricky with variables

`'a = int -> 'b`

`'a = 'c -> int`

Charles University

TinyHM

Constraint generation

- Generate type and constraints recursively
- Generate new fresh type variables as needed
- Variables with new type variables in context
- Most checking done via constraints

Charles University

Sketch

Generating constraints