Chapitre 4 Inférence de types

Programmation fonctionnelle, Licence d'informatique, 2021

Mini ML

Notons a, a1, a2... les expressions de mini ML

Exemple d'expression

```
fun x \rightarrow + (x, 1) abstraction de fonction
            + (x, 1) application d'une fonction
+ opération primitive (x, 1) construction d'une paire
                 x variable
                                1 constante
```

Types de mini ML

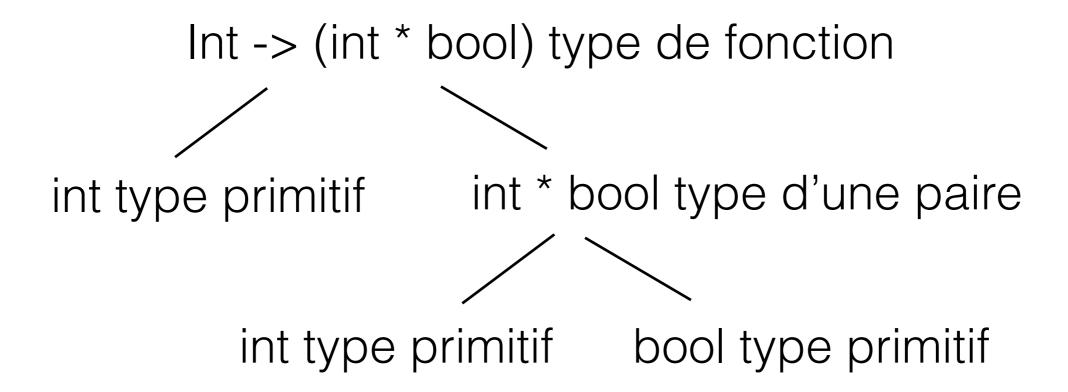
Notons t, t1, t2... les types

```
t := T type primitif (int, bool, char...)

| 'a variable de type

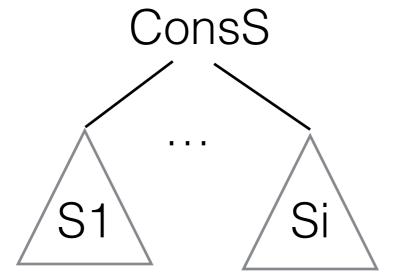
| t1 -> t2 type des fonctions de t1 vers t2

| t1 * t2 type des paires de t1 et t2
```

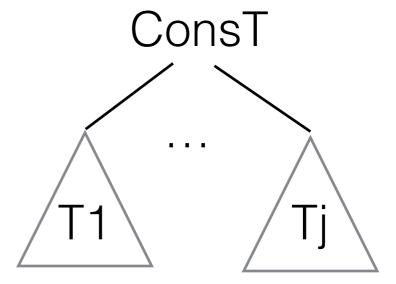


Unification

Terme S



Terme T



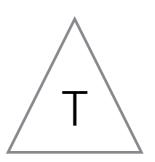
S et T s'unifient ssi si ConsS = ConsT (et donc i=j), et S1 unifie T1 et ... et Si unifie Tj sinon, échec de l'unification

Unification

Variable S

Terme T

S



L'unification est en succès et S <- T

Variable S

Variable T

S

L'unification est en succès et S <- U et T <- U (où U est une nouvelle variable)

Exemple

- mgu('a, int) est la substitution ['a <- int]
- mgu('a->'a, 'b->'int) est ['a <- 'c, 'b <- 'c], ['c <- int]
- mgu('a->'a, int->bool) échoue, car int et bool ne peuvent être unifiés

Exercice

- Calculer les unifications suivantes :
 - mgu('a->('b->'a), 'c->int)
 - mgu('a->'b, 'c*'d)
 - mgu(int*char->'a, 'b->bool)

Correction

- Calculer les unifications suivantes :
 - mgu('a->('b->'a), 'c->int) ['a <- 'd, 'c <- 'd],
 Echec
 - mgu('a->'b, 'c*'d) Echec
 - mgu(int*char->'a, 'b->bool) ['b <- (int * char)], ['a <- bool]

Règles de typage monomorphe

```
E \cup \{x : t\} \mid -x : t \text{ (var)}
                                         E \cup \{x : t1\} \mid -a : t2
                                                             — (fun)
  E |- op : t_op (op)
                                      E \mid - (fun x -> a) : t1 -> t2
  E \mid -c:t\_c (const)
                                   E \mid -a1:t1->t2 \quad E \mid -a2:t1
                                                                -- (app)
E |- a1 : t1 E |- a2 : t2
                                            E |- a1 a2 : t2
                ----- (paire)
 E |- (a1, a2) : t1 * t2
                                E |- a1 : t1 E U {x : t1} |- a2 : t2
                                                                     - (let)
                                     E \mid - (let x = a1 in a2) : t2
```

(6)
$$|-(fun x -> + (x, 1)) : 'a$$

$$(5) \{x : 'b\} | - + (x, 1) : 'c$$

(6)
$$|-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))$$

$$(5) \{x : 'b\} | - + (x, 1) : 'c$$

(6)
$$|-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))$$

- $(4) \{x : 'b\} | (x, 1) : 'd$
- (5) $\{x : 'b\} \mid -+ (x, 1) : 'c (app (1) et (4))$
- (6) |-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))

- $(4) \{x : 'b\} | (x, 1) : 'd$
- (5) $\{x : 'b\} \mid -+ (x, 1) : 'c (app (1) et (4)) \}$
- (6) |-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))

 $E \mid -c:t_c (const)$

t c = int

mgu(f, int) = [f <- int]

(1)
$$\{x : 'b\} \mid -+: 'e * 'f -> 'c$$

$$(3) \{x : 'b\} | -1 : 'f$$

(5)
$$\{x : 'b\} \mid -+ (x, 1) : 'c (app (1) et (4))$$

(6)
$$|-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))$$

 $E \mid -c:t_c (const)$

t c = int

- (1) $\{x : 'b\} \mid -+: 'e * int-> 'c$
- $(2) \{x : 'b\} | x : 'e$
- $(3) \{x : 'b\} | -1 : int (const)$
- (4) {x : 'b} |- (x, 1) : 'e * int (paire (2) et (3))
- (5) $\{x : 'b\} \mid -+ (x, 1) : 'c (app (1) et (4))$
- (6) |-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))

 $(1) \{x : b\} \mid -+: e * int-> c$ $E \cup \{x : t\} \mid -x : t \text{ (var)}$ t = b $(2) \{x : 'b\} | - x : 'e$ mgu('b, 'e) = ['b < - 'g, 'e < - 'g](3) {x : 'b} |- 1 : int (const) $(4) \{x : b\} \mid -(x, 1) : e * int (paire (2) et (3))$ (5) $\{x : b\} \mid -+(x, 1) : c (app (1) et (4))$ (6) |-(fun x -> + (x, 1)) : 'b -> 'c (fun (5))

(1)
$$\{x : 'g\} \mid - + : 'g * int-> 'c$$
 $E \cup \{x : t\} \mid - x : t \text{ (var)}$
(2) $\{x : 'g\} \mid - x : 'g \text{ (var)}$

- (3) {x : 'g} |- 1 : int (const)
- (4) {x : 'g} |- (x, 1) : 'g * int (paire (2) et (3))
- (5) $\{x : 'g\} \mid -+ (x, 1) : 'c (app (1) et (4))$
- (6) |-(fun x -> + (x, 1)) : 'g -> 'c (fun (5))

```
(1) \{x : 'g\} \mid -+: 'g * int-> 'c
                                             E |- op : t_op (op)
(2) \{x : 'g\} \mid -x : 'g (var)
                                          t_op = int * int -> int
                                   mgu('g * int -> 'c, int * int -> int)
(3) \{x : 'g\} \mid -1 : int (const)
                                          = ['g <- int, 'c <- int]
(4) {x : 'g} |- (x, 1) : 'g * int (paire (2) et (3))
(5) \{x : 'g\} \mid -+ (x, 1) : 'c (app (1) et (4)) \}
(6) |-(fun x -> + (x, 1)) : 'g -> 'c (fun (5))
```

```
(1) \{x : int\} \mid -+: int * int-> int (op)
                                             E |- op : t_op (op)
(2) \{x : int\} \mid -x : int (var)
                                          t_op = int * int -> int
(3) {x : int} |- 1 : int (const)
(4) {x : int} |- (x, 1) : int * int (paire (2) et (3))
(5) \{x : int\} \mid -+(x, 1) : int (app (1) et (4))
(6) |-(fun x -> + (x, 1)) : int -> int (fun (5))
```

- (1) $\{x : int\} \mid -+: int * int-> int (op)$
- (2) $\{x : int\} \mid -x : int (var)$
- (3) {x : int} |- 1 : int (const)
- (4) {x : int} |- (x, 1) : int * int (paire (2) et (3))
- (5) $\{x : int\} \mid -+(x, 1) : int (app (1) et (4))$
- (6) |-(fun x -> + (x, 1)) : int -> int (fun (5))

$$E \mid -a1 : t1 \quad E \cup \{x : t1\} \mid -a2 : t2$$
 $E \mid -(let x = a1 in a2) : t2$
 $E \mid -(b in a2) : t2$

(9) |- let
$$f = \text{fun } x -> x \text{ in } (f 1, f \text{ true})$$
: 'a

```
(2) |- fun x -> x : 'b
E |- a1 : t1 \quad E \cup \{x : t1\} |- a2 : t2
E |- (let x = a1 in a2) : t2
t2 = 'a, t1 = 'b
```

- (8) {f: 'b} |- (f 1, f true): 'a
- (9) |- let f = fun x -> x in (f 1, f true) : 'a (let (2) et (8))

```
(2) |- \text{ fun } x -> x : 'b
```

```
(8) {f: 'b} |- (f 1, f true): 'a
```

(9)
$$|-|$$
 let $f = fun x -> x in (f 1, f true) : 'a (let (2) et (8))$

```
(2) |- fun x -> x : 'b
```

- (7) {f: 'b} |- f true: 'd
- (8) {f: 'b} |- (f 1, f true): 'c * 'd (paire (5) et (7))
- (9) |- let f = fun x -> x in (f 1, f true) : 'c * 'd (let (2) et (8))

```
(2) |- fun x -> x : 'b
```

- (5) {f: 'b} |-f1: 'c
- (7) {f: 'b} |- f true: 'd
- (8) {f: 'b} |- (f 1, f true): 'c * 'd (paire (5) et (7))
- (9) |- let f = fun x -> x in (f 1, f true) : 'c * 'd (let (2) et (8))

```
E \mid -a1:t1->t2 \quad E \mid -a2:t1
                                                                -- (app)
(2) \mid- fun x -> x : 'b
                                            E |- a1 a2 : t2
(3) \{f : 'b\} \mid -f : 'e -> 'd
                                           t1 = e, t2 = d
(5) {f: 'b} |- f 1: 'c
(6) {f: 'b} |- true: 'e
(7) {f: 'b} |- f true: 'd (app (3) et (6))
```

(9) |- let $f = fun \times -> x in (f 1, f true) : 'c * 'd (let (2) et (8))$

(8) {f: 'b} |- (f 1, f true): 'c * 'd (paire (5) et (7))

```
E \mid -c:t\_c (const)
(2) \mid- fun x -> x : 'b
(3) \{f : 'b\} \mid -f : 'e -> 'd
                                              t c = bool
                                   mgu(e, bool) = [e < -bool]
(5) {f: 'b} |- f 1: 'c
(6) {f: 'b} |- true: 'e
(7) {f: 'b} |- f true: 'd (app (3) et (6))
(8) {f: 'b} |- (f 1, f true): 'c * 'd (paire (5) et (7))
(9) |- let f = fun \times -> x in (f 1, f true) : 'c * 'd (let (2) et (8))
```

```
E \mid -c:t\_c (const)
(2) \mid- fun x -> x : 'b
(3) \{f : b\} \mid -f : bool -> d
                                              t c = bool
(5) {f: 'b} |- f 1: 'c
(6) {f: 'b} |- true: bool (const)
(7) {f: 'b} |- f true: 'd (app (3) et (6))
(8) {f: 'b} |- (f 1, f true): 'c * 'd (paire (5) et (7))
(9) |- let f = fun \times -> x in (f 1, f true) : 'c * 'd (let (2) et (8))
```

```
E \mid -a1:t1->t2 \quad E \mid -a2:t1
                                                              -- (app)
(2) \mid- fun x -> x : 'b
                                           E |- a1 a2 : t2
(3) \{f : b\} \mid -f : bool -> d
                                        t1 = bool, t2 = c
                               mgu('d, 'c) = ['d <- 'e, 'c <- 'e]
(5) {f: 'b} |- f 1: 'c
(6) {f: 'b} |- true: bool (const)
(7) {f: 'b} |- f true: 'd (app (3) et (6))
(8) {f: 'b} |- (f 1, f true): 'c * 'd (paire (5) et (7))
(9) |- let f = fun \times -> x in (f 1, f true) : 'c * 'd (let (2) et (8))
```

```
E \mid -a1:t1->t2 \quad E \mid -a2:t1
                                                              -- (app)
(2) \mid- fun x -> x : 'b
                                           E |- a1 a2 : t2
(3) \{f : 'b\} | - f : bool -> 'e
                                        t1 = bool, t2 = c
(4) {f: 'b} |- 1: bool
(5) {f: 'b} |- f 1: 'e (app (3) et (4))
(6) {f: 'b} |- true: bool (const)
(7) {f: 'b} |- f true: 'e (app (3) et (6))
(8) {f: 'b} |- (f 1, f true): 'e * 'e (paire (5) et (7))
(9) |- let f = fun \times -> x in (f 1, f true) : 'e * 'e (let (2) et (8))
```

```
E \mid -c:t\_c (const)
(2) \mid- fun x -> x : 'b
                                                 t c = int
(3) \{f : 'b\} \mid -f : bool -> 'e
                                       mgu (bool, int) = Echec!
(4) {f: 'b} |- 1: bool (const)
(5) {f: 'b} |- f 1: 'e (app (3) et (4))
(6) {f: 'b} |- true: bool (const)
(7) {f: 'b} |- f true: 'e (app (3) et (6))
(8) {f: 'b} |- (f 1, f true): 'e * 'e (paire (5) et (7))
(9) |- let f = fun \times -> x in (f 1, f true) : 'e * 'e (let (2) et (8))
```

Exercices

- Inférer les types des expressions suivantes :
 - fun x -> (&& (x, true))
 - fun f -> (f f)

(6)
$$\mid$$
- fun x -> (&& (x, true)) : 'a

- (5) {x: 'b} |- && (x, true): 'c
- (6) \mid fun x -> (&& (x, true)) : 'b -> 'c (fun (5))

- (5) {x: 'b} |- && (x, true): 'c
- (6) \mid fun x -> (&& (x, true)) : 'b -> 'c (fun (5))

```
(1) \{x : 'b\} | - \&\& : 'd -> 'c
```

(4)
$$\{x : 'b\} \mid -(x, true) : 'd$$
 $t1 = 'd, t2 = 'c$

(6)
$$\mid$$
- fun x -> (&& (x, true)) : 'b -> 'c (fun (5))

- (4) {x : 'b} |- (x, true) : 'd
- (5) {x : 'b} |- && (x, true) : 'c (app (1) et (4))
- (6) \mid fun x -> (&& (x, true)) : 'b -> 'c (fun (5))

```
E |- a1 : t1 E |- a2 : t2
(1) \{x : 'b\} \mid - \&\& : 'e * 'f -> 'c
                                                        -- (paire)
                                       E |- (a1, a2) : t1 * t2
(2) \{x : 'b\} | - x : 'e
                                           t1 = e, t2 = f
(3) {x : 'b} |- true : 'f
(4) {x : 'b} |- (x, true) : 'e * 'f (paire (2) et (3))
(5) {x : 'b} |- && (x, true) : 'c (app (1) et (4))
(6) |- fun x -> (&& (x, true)) : 'b -> 'c (fun (5))
```

```
(1) \{x : b\} \mid - \&\& : e * f -> c
                                            E \mid -c:t\_c (const)
                                             t c = bool
(2) \{x : 'b\} | - x : 'e
                                   mgu(f, bool) = [f <- bool]
(3) {x : 'b} |- true : 'f
(4) {x : 'b} |- (x, true) : 'e * 'f (paire (2) et (3))
(5) {x : 'b} |- && (x, true) : 'c (app (1) et (4))
(6) |- fun x -> (&& (x, true)) : 'b -> 'c (fun (5))
```

- (4) {x : 'b} |- (x, true) : 'e * bool (paire (2) et (3))
- (5) {x: 'b} |- && (x, true): 'c (app (1) et (4))
- (6) \mid fun x -> (&& (x, true)) : 'b -> 'c (fun (5))

```
(1) \{x : b\} \mid -\&\& : e * bool -> c \quad E \cup \{x : t\} \mid -x : t \quad (var) \in A
                                                 t = b
(2) \{x : 'b\} | - x : 'e
                                 mgu('e, 'b) = ['e < - 'g, 'b < - 'g]
(3) {x : 'b} |- true : bool (const)
(4) {x : 'b} |- (x, true) : 'e * bool (paire (2) et (3))
(5) {x : 'b} |- && (x, true) : 'c (app (1) et (4))
(6) |- fun x -> (&& (x, true)) : 'b -> 'c (fun (5))
```

```
(1) \{x : 'g\} \mid -\&\& : 'g * bool -> 'c \quad E \cup \{x : t\} \mid -x : t \ (var)\}
                                                 t = b
(2) \{x : 'g\} | -x : 'g (var)
(3) {x : 'g} |- true : bool (const)
(4) {x: 'g} |- (x, true): 'g * bool (paire (2) et (3))
(5) {x : 'g} |- && (x, true) : 'c (app (1) et (4))
(6) \mid- fun x -> (&& (x, true)) : 'g -> 'c (fun (5))
```

```
(1) \{x : 'g\} \mid - \&\& : 'g * bool -> 'c
                                             E |- op : t_op (op)
                                   t_op = bool * bool -> bool
(2) \{x : 'g\} \mid -x : 'g (var)
                                       mgu('g * bool ->'c,
                                             bool * bool -> bool) =
(3) {x : 'g} |- true : bool (const)
                                     ['g <- bool, 'c <- bool]
(4) {x : 'g} |- (x, true) : 'g * bool (paire (2) et (3))
(5) {x : 'g} |- && (x, true) : 'c (app (1) et (4))
(6) \mid- fun x -> (&& (x, true)) : 'g -> 'c (fun (5))
```

```
(1) \{x : bool\} \mid - \&\& : bool * bool -> bool (op)
                                            E |- op : t_op (op)
(2) \{x : bool\} | -x : bool (var) | t_op = bool * bool -> bool
(3) {x : bool} |- true : bool (const)
(4) {x : bool} |- (x, true) : bool * bool (paire (2) et (3))
(5) {x : bool} |- && (x, true) : bool (app (1) et (4))
(6) |- fun x -> (&& (x, true)) : bool -> bool (fun (5))
```

- (1) $\{x : bool\} \mid \&\& : bool * bool -> bool (op)$
- (2) {x : bool} |- x : bool (var)
- (3) {x : bool} |- true : bool (const)
- (4) {x : bool} |- (x, true) : bool * bool (paire (2) et (3))
- (5) {x : bool} |- && (x, true) : bool (app (1) et (4))
- (6) \mid fun x -> (&& (x, true)) : **bool -> bool** (fun (5))

$$(4) \mid - \text{fun } f \rightarrow (f f) : `a$$

- (3) {f: 'b} |- ff: 'c
- $(4) \mid \text{fun } f \rightarrow (f f) : 'b \rightarrow 'c (fun (3))$

- (3) {f: 'b} |- ff: 'c
- $(4) \mid \text{fun } f \rightarrow (f f) : 'b \rightarrow 'c (fun (3))$

```
E \mid -a1:t1->t2 \quad E \mid -a2:t1
                                                                    -- (app)
                                              E |- a1 a2 : t2
(1) \{f : 'b\} | -f : 'd -> 'c
                                              t1 = 'd, t2 = 'c
(2) {f: 'b} |- f: 'd
(3) {f: 'b} |- ff: 'c (app (1) et (2))
(4) \mid - \text{ fun } f \rightarrow (f f) : 'b \rightarrow 'c (fun (3))
```

```
E \cup \{x : t\} \mid -x : t \ (var)
                                                          t = b
(1) \{f : 'b\} \mid -f : 'd \rightarrow 'c
                                      mgu('d, 'b) = ['d <- 'e, 'b <- 'e]
(2) {f: 'b} |- f: 'd
(3) {f: 'b} |- ff: 'c (app (1) et (2))
(4) \mid - \text{ fun } f \rightarrow (f f) : 'b \rightarrow 'c (fun (3))
```

```
E \cup \{x : t\} \mid -x : t \text{ (var)}

t = b
```

- $(1) \{f : e\} | -f : e -> c$
- (2) {f: 'e} |- f: 'e (var)
- (3) {f: 'e} |- ff: 'c (app (1) et (2))
- $(4) \mid \text{fun } f \rightarrow (f f) : 'e \rightarrow 'c (fun (3))$

```
E \cup \{x : t\} \mid -x : t \text{ (var)}
                                                        t = e
(1) \{f : e\} \mid -f : e \rightarrow c
                                         mgu('e, 'e \rightarrow 'c) = Echec!
(2) {f: 'e} |- f: 'e (var)
(3) {f: 'e} |- ff: 'c (app (1) et (2))
(4) |- fun f -> (f f): 'e -> 'c (fun (3))
```

```
E \cup \{x : t\} \mid -x : t \ (var)
                                                             t = e
(1) \{f : e\} \mid f : e \rightarrow c \text{ (var) } mgu(e, e \rightarrow c) = Echec!
(2) {f: 'e} |- f: 'e (var)
(3) {f: 'e} |- ff: 'c (app (1) et (2))
(4) \mid - \text{fun } f \rightarrow (f f) : 'e \rightarrow 'c (fun (3))
```

Règles de typage polymorphe

```
E U \{x : t1\} \mid -a:t2
E \cup \{x : t\} \mid -x : t \text{ (var)}
                                                  ----- (fun)
  E |- op : t_op (op)
                                         E \mid - (fun x -> a):t1->t2
  E \mid -c:t\_c (const)
                                     E |- a1:t1->t2 E |- a2:t1
                                                             -(app)
E |- a1 : t1 E |- a2 : t2
                                             E |- a1 a2:t2
            ------ (paire)
 E |- (a1, a2) : t1 * t2
                                        E \mid -a2[x<-a1]:t2
                                                          -(let-sub)
                                    E \mid - (let x = a1 in a2):t2
```

(10) |-| let f = fun x -> x in (f 1, f true) : 'a

```
(9) |- ((fun x -> x) 1, (fun x -> x) true) : 'a
(10) |- let f = fun x -> x in (f 1, f true) : 'a (let-sub (9))
```

```
(4) | - (fun x -> x) 1 : 'b
```

- (8) |- (fun x -> x) true : 'c
- (9) |-((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))
- (10) |- let f = fun x -> x in (f 1, f true) : 'b * 'c (let-sub (9))

$$(4) | - (fun x -> x) 1 : 'b$$

(9)
$$|-((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))$$

(10) |- let
$$f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * 'c (let-sub (9))$$

- $(4) \mid (\text{fun } x -> x) 1 : 'b$
- (6) |- fun x -> x : 'd -> 'c
- (7) |- true: 'd
- (8) |-(fun x -> x) true : 'c (app (6) et (7))
- (9) [-((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))]
- (10) |- let f = fun x -> x in (f 1, f true): 'b * 'c (let-sub (9))

 $E \mid -c:t_c (const)$

```
t c = bool
                                     mqu('d, bool) = ['d <- bool]
(4) \mid - (\text{fun } x -> x) 1 : 'b
(6) \mid- fun x -> x : 'd -> 'c
(7) |- true : 'd
(8) |- (fun x -> x) true : 'c (app (6) et (7))
(9) -((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))
(10) |- let f = fun x -> x in (f 1, f true) : 'b * 'c (let-sub (9))
```

```
E \mid -c : t_c \text{ (const)}
t_c = bool
```

- (4) | (fun x -> x) 1 : 'b
- (6) |- fun x -> x : bool -> `c
- (7) |- true : bool (const)
- (8) \mid (fun x -> x) true : 'c (app (6) et (7))
- (9) |-((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))
- (10) |- let f = fun x -> x in (f 1, f true): 'b * 'c (let-sub (9))

$$(4) | - (fun x -> x) 1 : 'b$$

$$t1 = bool, t2 = c$$

- (6) |- fun x -> x : bool -> `c
- (7) |- true : bool (const)
- (8) |-(fun x -> x) true : 'c (app (6) et (7))
- (9) [-((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))]
- (10) |- let f = fun x -> x in (f 1, f true) : 'b * 'c (let-sub (9))

t1 = bool, t2 = c

- (4) | (fun x -> x) 1 : 'b
- $(5) \{x : bool\} | -x : `c$
- (6) \mid fun x -> x : bool -> 'c (fun (5))
- (7) |- true : bool (const)
- (8) \mid (fun x -> x) true : 'c (app (6) et (7))
- (9) $\left| ((\text{fun } x -> x) \ 1, (\text{fun } x -> x) \ \text{true}) \right| : 'b * 'c (paire (4) et (8))$
- (10) |- let f = fun x -> x in (f 1, f true) : 'b * 'c (let-sub (9))

```
E \cup \{x : t\} \mid -x : t \text{ (var)}
                                                    t = bool
                                       mgu(c, bool) = [c < -bool]
(4) \mid - (\text{fun } x -> x) 1 : 'b
(5) \{x : bool\} | -x : `c
(6) \mid- fun x -> x : bool -> 'c (fun (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : 'c (app (6) et (7))
(9) -((fun x -> x) 1, (fun x -> x) true) : 'b * 'c (paire (4) et (8))
(10) |- let f = fun x -> x in (f 1, f true) : 'b * 'c (let-sub (9))
```

```
E \cup \{x : t\} \mid -x : t \text{ (var)}t = bool
```

- $(4) \mid (fun x -> x) 1 : 'b$
- $(5) \{x : bool\} | -x : bool (var)$
- (6) \mid fun x -> x : bool -> bool (fun (5))
- (7) |- true : bool (const)
- (8) \mid (fun x -> x) true : bool (app (6) et (7))
- (9) |- ((fun x -> x) 1, (fun x -> x) true): 'b * bool (paire (4) et (8))
- (10) |- let f = fun x -> x in (f 1, f true) : 'b * bool (let-sub (9))

t1 = 'd, t2 = 'b

- $(4) \mid (fun x -> x) 1 : 'b$
- (5) {x : bool} |- x : bool (var)
- (6) |- fun x -> x : bool -> bool (fun (5))
- (7) |- true : bool (const)
- (8) |- (fun x -> x) true : bool (app (6) et (7))
- (9) |- ((fun x -> x) 1, (fun x -> x) true): 'b * bool (paire (4) et (8))
- (10) |- let f = fun x -> x in (f 1, f true) : 'b * bool (let-sub (9))

```
E \mid -a1:t1->t2 \quad E \mid -a2:t1
                                                                               -(app)
(2) \mid - \text{fun } x -> x : 'd -> 'b'
                                                          E |- a1 a2:t2
(3) |- 1 : 'd
(4) |- (fun x -> x) 1 : 'b (app (2) et (3)) t1 = 'd, t2 = 'b
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
(9) |- ((fun x -> x) 1, (fun x -> x) true) : 'b * bool (paire (4) et (8))
(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * bool (let-sub (9))
```

```
E \mid -c:t\_c (const)
(2) \mid - \text{fun } x -> x : 'd -> 'b'
                                                                  t c = int
                                                     mgu('d, int) = ['d <- int]
(3) |- 1 : 'd
(4) \mid - (\text{fun } x -> x) \ 1 : \text{'b (app (2) et (3))}
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
(9) |- ((fun x -> x) 1, (fun x -> x) true) : 'b * bool (paire (4) et (8))
(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * bool (let-sub (9))
```

```
E \mid -c:t\_c (const)
(2) \mid- fun x -> x : int -> 'b
                                                                 t c = int
(3) |- 1 : int (const)
(4) \mid - (\text{fun } x -> x) \ 1 : 'b (app (2) et (3))
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
(9) |- ((fun x -> x) 1, (fun x -> x) true) : 'b * bool (paire (4) et (8))
(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * bool (let-sub (9))
```

```
(2) \mid- fun x -> x : int -> 'b
(3) |- 1 : int (const)
(4) \mid - (\text{fun } x -> x) \ 1 : \text{'b (app (2) et (3))}
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
```

```
E \cup \{x : t1\} \mid -a:t2
                                                             —— (fun)
                                             E \mid - (fun x -> a):t1->t2
                                                 t1 = int, t2 = 'b
(9) |- ((fun x -> x) 1, (fun x -> x) true) : 'b * bool (paire (4) et (8))
```

 $(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * bool (let-sub (9))$

 $E \cup \{x : t1\} \mid -a:t2$ $(1) \{x : int\} \mid -x : 'b$ —— (fun) $(2) \mid - \text{ fun } x -> x : \text{ int } -> \text{ 'b (fun (1))}$ $E \mid - (fun x -> a):t1->t2$ (3) |- 1 : int (const) t1 = int, t2 = 'b $(4) \mid - (\text{fun } x -> x) \ 1 : 'b (app (2) et (3))$ (5) {x : bool} |- x : bool (var) (6) |-fun x -> x : bool -> bool (fun (5))(7) |- true : bool (const) (8) |- (fun x -> x) true : bool (app (6) et (7)) (9) |- ((fun x -> x) 1, (fun x -> x) true) : 'b * bool (paire (4) et (8))

 $(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * bool (let-sub (9))$

```
E \cup \{x : t\} \mid -x : t \ (var)
(1) \{x : int\} \mid -x : 'b
                                                                      t = int
(2) \mid - \text{ fun } x -> x : \text{ int } -> \text{ 'b (fun (1))}
                                                       mgu(b, int) = [b <- int]
(3) |- 1 : int (const)
(4) \mid - (\text{fun } x -> x) \ 1 : 'b (app (2) et (3))
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
(9) |- ((fun x -> x) 1, (fun x -> x) true) : 'b * bool (paire (4) et (8))
(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : 'b * bool (let-sub (9))
```

```
E \cup \{x : t\} \mid -x : t \ (var)
(1) \{x : int\} \mid -x : int (var)
                                                                           t = int
(2) \mid - \text{ fun } x -> x : \text{ int } -> \text{ int } (\text{fun } (1))
(3) |- 1 : int (const)
(4) \mid - (\text{fun } x -> x) \ 1 : 'b (app (2) et (3))
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
(9) |- ((fun x -> x) 1, (fun x -> x) true) : int * bool (paire (4) et (8))
(10) \mid - \text{let } f = \text{fun } x -> x \text{ in } (f 1, f \text{ true}) : \text{int } * \text{bool } (\text{let-sub } (9))
```

```
(1) \{x : int\} \mid -x : int (var)
(2) \mid - \text{ fun } x -> x : \text{ int } -> \text{ int } (\text{fun } (1))
(3) |- 1 : int (const)
(4) \mid - (\text{fun } x -> x) \ 1 : \text{'b (app (2) et (3))}
(5) {x : bool} |- x : bool (var)
(6) |-\text{fun } x -> x : \text{bool } -> \text{bool } (\text{fun } (5))
(7) |- true : bool (const)
(8) |- (fun x -> x) true : bool (app (6) et (7))
(9) |- ((fun x -> x) 1, (fun x -> x) true) : int * bool (paire (4) et (8))
(10) |- let f = fun x -> x in (f 1, f true) : int * bool (let-sub (9))
```

Exercices

- Inférer les types l'expression suivante :
 - let x = (1, true) in (fun y -> x)

(5)
$$|-|$$
 let $x = (1, true)$ in (fun $y -> x$): 'a

- $(4) \mid \text{ fun y } -> (1, \text{ true}) : `a$
- (5) |-| let x = (1, true) in (fun y -> x): 'a (let-sub (4))

- $(4) \mid \text{ fun y } -> (1, \text{ true}) : `a$
- (5) |-| let x = (1, true) in (fun y -> x): 'a (let-sub (4))

- (3) {y: 'b} |- (1, true): 'c
- (4) |- fun y -> (1, true): 'b -> 'c (fun (3))
- (5) |-| let x = (1, true) in (fun y -> x): 'b -> 'c (let-sub (4))

- (3) {y: 'b} |- (1, true): 'c
- (4) |- fun y -> (1, true): 'b -> 'c (fun (3))
- (5) |-| let x = (1, true) in (fun y -> x): 'b -> 'c (let-sub (4))

```
E |- a1 : t1 E |- a2 : t2
                                                    ----- (paire)
(1) {y: 'b} |- 1: 'd
                                      E |- (a1, a2) : t1 * t2
(2) {y: 'b} |- true: 'e
                                         t1 = 'd, t2 = 'e
(3) {y: 'b} |- (1, true): 'd * 'e (paire (1) et (2))
(4) |- fun y -> (1, true): 'b -> 'd * 'e (fun (3))
(5) |-| let x = (1, true) in (fun y -> x):
                                    'b -> 'd * 'e (let-sub (4))
```

```
E \mid -c:t\_c (const)
(1) {y: 'b} |- 1: 'd
                                              t c = bool
                                   mgu(e, bool) = [e <- bool]
(2) {y: 'b} |- true: 'e
(3) {y: 'b} |- (1, true): 'd * 'e (paire (1) et (2))
(4) |- fun y -> (1, true): 'b -> 'd * 'e (fun (3))
(5) |- \text{ let } x = (1, \text{ true}) \text{ in (fun } y -> x) :
                                      'b -> 'd * 'e (let-sub (4))
```

```
E \mid -c:t\_c (const)
(1) {y: 'b} |- 1: 'd
                                          t_c = bool
(2) {y : 'b} |- true : bool (const)
(3) {y : 'b} |- (1, true) : 'd * bool (paire (1) et (2))
(4) |- fun y -> (1, true): 'b -> 'd * bool (fun (3))
(5) |-| let x = (1, true) in (fun y -> x):
                                 'b -> 'd * bool (let-sub (4))
```

```
E \mid -c:t\_c (const)
(1) {y: 'b} |- 1: 'd
                                              t c = int
                                     mgu('d, int) = ['d <- int]
(2) {y : 'b} |- true : bool (const)
(3) {y : 'b} |- (1, true) : 'd * bool (paire (1) et (2))
(4) |- fun y -> (1, true): 'b -> 'd * bool (fun (3))
(5) |-| let x = (1, true) in (fun y -> x):
                                 'b -> 'd * bool (let-sub (4))
```

```
E \mid -c:t\_c (const)
(1) {y: 'b} |- 1: int
                                                  t_c c = int
(2) {y : 'b} |- true : bool (const)
(3) {y: 'b} |- (1, true): int * bool (paire (1) et (2))
(4) |- fun y -> (1, true): 'b -> int * bool (fun (3))
(5) |- \text{ let } x = (1, \text{ true}) \text{ in (fun } y -> x) :
                                   'b -> int * bool (let-sub (4))
```

(1) {y: 'b} |- 1: int (2) {y : 'b} |- true : bool (const) (3) {y : 'b} |- (1, true) : int * bool (paire (1) et (2)) $(4) \mid - \text{ fun y } -> (1, \text{ true}) : \text{ 'b } -> \text{ int * bool (fun (3))}$ (5) |-| let x = (1, true) in (fun y -> x): **'b -> int * bool** (let-sub (4))