

Paradigmes et Interprétation

Sous-typage

Julien Provillard

julien.provillard@univ-cotedazur.fr

TYPAGE DES ENREGISTREMENTS

Grammaire

```
<Exp> ::= <Number>
        | <Symbol>
        | {+ <Exp> <Exp>}
        | {* <Exp> <Exp>}
        | {lambda {[<Symbol> : <Type>]} <Exp>}
        | {<Exp> <Exp>}
        | {record [<Symbol> <Exp>]*}
        | {get <Exp> <Symbol>}
        | {set <Exp> <Symbol> <Exp>}
```

```
<Type> ::= num
        | bool
        | (<Type> -> <Type>)
        | {[<Symbol> : <Type>]*}
```

Typage des enregistrements

□ Le type de

```
{record  
  [x {+ 1 2}]  
  [y {* 3 4}]}
```

est

```
{[x : num] [y : num]}
```

Typage des enregistrements

□ Le type de

```
{get {record  
    [x {+ 1 2}]  
    [y {* 3 4}]}  
x}
```

est

num

Typage des enregistrements

□ Le type de

```
{get
  {get
    {record
      [p {record
        [x {+ 1 2}]
        [y {* 3 4}]]]]}
  p}
x}
```

est

num

Typage des enregistrements

□ Le type de

```
{set {record  
    [x {+ 1 2}]  
    [y {* 3 4}]}}
```

x

5}

est

?

Typage des enregistrements

□ Le type de

```
{set {record
  [x {+ 1 2}]
  [y {* 3 4}]}}
x
5}
```

est le même que la valeur produite

```
{record
  [x 5]
  [y 12]}
```

soit

```
{[x : num] [y : num]}
```


Typage des enregistrements

□ Le type de

```
{{lambda {[r : {[x : num] [y : num]}]}  
  {+ {get r x} {get r y}}}  
{record  
  [x {+ 1 2}]  
  [y {* 3 4}]}}
```

est

num

Typage des enregistrements

□ Le type de

```
{lambda {[r : {[x : num] [y : num]}]}  
  {set r x {get r y}}}
```

est

```
({[x : num] [y : num]} -> {[x : num] [y : num]})
```

Règles de typage

$$\begin{array}{c}
 \Gamma \vdash \text{expr}_1 : \tau_1 \quad \dots \quad \Gamma \vdash \text{expr}_n : \tau_n \\
 \tau = \{[x_1 : \tau_1] \quad \dots \quad [x_n : \tau_n]\} \\
 \hline
 \Gamma \vdash \{\text{record } [x_1 \text{ expr}_1] \quad \dots \quad [x_n \text{ expr}_n]\} : \tau
 \end{array}$$

$$\begin{array}{c}
 \Gamma \vdash \text{expr} : \{[x_1 : \tau_1] \quad \dots \quad [x_n : \tau_n]\} \\
 x = x_i \\
 \hline
 \Gamma \vdash \{\text{get expr } x\} : \tau_i
 \end{array}$$

$$\begin{array}{c}
 \Gamma \vdash \text{expr}_1 : \{[x_1 : \tau_1] \quad \dots \quad [x_n : \tau_n]\} \\
 x = x_i \quad \Gamma \vdash \text{expr}_2 : \tau_i \\
 \hline
 \Gamma \vdash \{\text{set expr}_1 \ x \ \text{expr}_2\} : \{[x_1 : \tau_1] \quad \dots \quad [x_n : \tau_n]\}
 \end{array}$$

SOUS-TYPAGE

Typage, enregistrements et champs

$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num})$

$\{\{\text{lambda } \{[r : \{[x : \text{num}] [y : \text{num}]\}]\}$
 $\{\text{get } r \ x\}\}$

$\{\text{record } [x \ 1] [y \ 2]\}\}$

$\{[x : \text{num}] [y : \text{num}]\}$

L'expression a pour type num .

$$\frac{\Gamma \vdash \text{expr}_1 : (\tau_1 \rightarrow \tau_2) \quad \Gamma \vdash \text{expr}_2 : \tau_1}{\Gamma \vdash \{\text{expr}_1 \ \text{expr}_2\} : \tau_2}$$

Typage, enregistrements et champs

$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num})$

$\{\{\text{lambda } \{[r : \{[x : \text{num}] [y : \text{num}]\}]\}$
 $\{\text{get } r \ x\}\}$

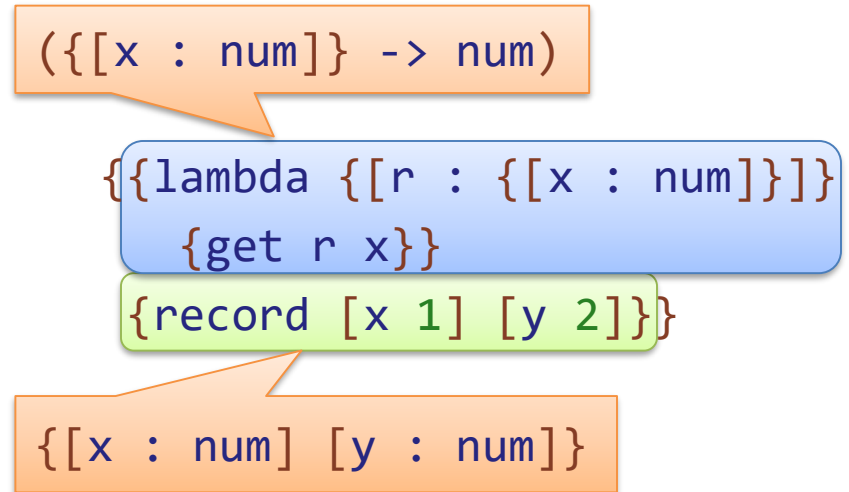
$\{\text{record } [y \ 1] [x \ 2]\}\}$

$\{[y : \text{num}] [x : \text{num}]\}$

Non-typable : L'ordre des champs ne correspond pas.

$$\frac{\Gamma \vdash \text{expr}_1 : (\tau_1 \rightarrow \tau_2) \quad \Gamma \vdash \text{expr}_2 : \tau_1}{\Gamma \vdash \{\text{expr}_1 \ \text{expr}_2\} : \tau_2}$$

Typage, enregistrements et champs



Non-typable : L'argument a des champs supplémentaires.

$$\frac{\Gamma \vdash \text{expr}_1 : (\tau_1 \rightarrow \tau_2) \quad \Gamma \vdash \text{expr}_2 : \tau_1}{\Gamma \vdash \{\text{expr}_1 \ \text{expr}_2\} : \tau_2}$$

Sous-types

□ Un type τ est un sous-type du type τ' (on note $\tau \leq \tau'$) lorsqu'une expression du type τ peut être utilisée à la place d'une expression du type τ' .

$$\{[x : \text{num}] [y : \text{num}]\} \leq \{[x : \text{num}]\}$$

$$\{[y : \text{num}] [x : \text{num}]\} \leq \{[x : \text{num}] [y : \text{num}]\}$$

$$\{[x : \text{num}]\} \leq \{[x : \text{num}]\}$$

$$\{[x : \text{num}]\} \not\leq \{[x : \text{num}] [y : \text{num}]\}$$

Règles pour le sous-typage

$$\frac{\begin{array}{c} \{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\} \\ x_i = x'_j \Rightarrow \tau_i = \tau'_j \end{array}}{\{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}}$$

$$\text{num} \leq \text{num}$$

$$\text{bool} \leq \text{bool}$$

$$(\tau_1 \rightarrow \tau_2) \leq (\tau_1 \rightarrow \tau_2)$$

$$\frac{\Gamma \vdash \text{expr}_1 : (\tau_1 \rightarrow \tau_2) \quad \Gamma \vdash \text{expr}_2 : \tau_3 \quad \tau_3 \leq \tau_1}{\Gamma \vdash \{\text{expr}_1 \text{ expr}_2\} : \tau_2}$$

Typage, enregistrements et champs

$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num})$

$\{\{\text{lambda } \{[r : \{[x : \text{num}] [y : \text{num}]\}]\}$
 $\{\text{get } r \ x\}\}$

$\{\text{record } [y \ 1] [x \ 2]\}\}$

$\{[y : \text{num}] [x : \text{num}]\}$

L'expression a pour type **num** car

$\{[y : \text{num}] [x : \text{num}]\} \leq \{[x : \text{num}] [y : \text{num}]\}$

Typage, enregistrements et champs

$(\{[x : \text{num}]\} \rightarrow \text{num})$

$\{\{\text{lambda } \{[r : \{[x : \text{num}]\}]\}$
 $\{\text{get } r \ x\}\}$

$\{\text{record } [x \ 1] \ [y \ 2]\}\}$

$\{[x : \text{num}] \ [y : \text{num}]\}$

L'expression a pour type **num** car

$\{[x : \text{num}] \ [y : \text{num}]\} \leq \{[x : \text{num}]\}$

Typage, enregistrements et champs

$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num})$

$\{\{\text{lambda } \{[r : \{[x : \text{num}] [y : \text{num}]\}]\}$
 $\{\text{get } r \ x\}\}$

$\{\text{record } [x \ 1]\}\}$

$\{[x : \text{num}]\}$

Non-typable car

$\{[x : \text{num}]\} \not\leq \{[x : \text{num}] [y : \text{num}]\}$

Typage, enregistrements et champs

```
(([p : {[x : num]}] -> num)
```

```
{lambda {[r : {[p : {[x : num]}]}]  
  {get {get r p} x}}
```

```
{record [p {record [x 1] [y 2]}]}
```

```
{[p : {[x : num] [y : num]}]}
```

Non-typable car

$$\{[p : \{[x : \text{num}] [y : \text{num}]]\} \not\subseteq \{[p : \{[x : \text{num}]]\}$$

Est-ce ce que l'on attend ?

Règles pour le sous-typage

$$\{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\}$$

$$x_i = x'_j \Rightarrow \tau_i = \tau'_j$$

$$\{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}$$

$$\{[x : \text{num}] [y : \text{num}]\} \neq \{[x : \text{num}]\}$$

$$\Rightarrow$$

$$\{[p : \{[x : \text{num}] [y : \text{num}]\}]\} \not\leq \{[p : \{[x : \text{num}]\}]\}$$

Règles pour le sous-typage

$$\begin{array}{c}
 \{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\} \\
 x_i = x'_j \Rightarrow \tau_i \leq \tau'_j \\
 \hline
 \{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}
 \end{array}$$

$$\{[x : \text{num}] [y : \text{num}]\} \leq \{[x : \text{num}]\}$$

$$\Rightarrow$$

$$\{[p : \{[x : \text{num}] [y : \text{num}]\}]\} \leq \{[p : \{[x : \text{num}]\}]\}$$

Le sous-typage doit être récursif !

Sous-typage et mise à jour des champs

$$\{\text{set } \{\text{record } \{p \ \{\text{record } \{x \ 1\}\}\}\}\}$$

$$p$$

$$\{\text{record } \{x \ 1\} \ \{y \ 2\}\}$$

❑ Non-typable avec la règle actuelle.

$$\frac{\begin{array}{c} \Gamma \vdash \text{expr}_1 : \{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \\ x = x_i \quad \Gamma \vdash \text{expr}_2 : \tau_i \end{array}}{\Gamma \vdash \{\text{set } \text{expr}_1 \ x \ \text{expr}_2\} : \{[x_1 : \tau_1] \dots [x_n : \tau_n]\}}$$

❑ Règle modifiée

$$\frac{\begin{array}{c} \Gamma \vdash \text{expr}_1 : \{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \\ x = x_i \quad \Gamma \vdash \text{expr}_2 : \tau \quad \tau \leq \tau_i \end{array}}{\Gamma \vdash \{\text{set } \text{expr}_1 \ x \ \text{expr}_2\} : \{[x_1 : \tau_1] \dots [x_n : \tau_n]\}}$$

SOUS-TYPAGE ET FONCTIONS

Sous-typage et fonctions : Exemples

Je voudrais changer de portable !

Pourquoi pas un smartphone ?

`{[val : num]}`

`{[val : num] [gen : num]}`

OK

$\{[val : num] [gen : num]\} \leq \{[val : num]\}$

Sous-typage et fonctions : Exemple 1

Je connais un bon magasin de smartphones.

Tu l'achèterais où ton portable ?

`(num -> {[val : num]})`

`(num -> {[val : num] [gen : num]})`

OK

$(\text{num} \rightarrow \{[\text{val} : \text{num}] [\text{gen} : \text{num}]\}) \leq (\text{num} \rightarrow \{[\text{val} : \text{num}]\})$

Sous-typage et fonctions : Exemple 1

```
((num -> {[x : num]}) -> num)
```

```
{[lambda {[f : (num -> {[x : num]})]}]  
  {get {f 2} x}}
```

```
{lambda {[n : num]}  
  {record [x n] [y n]}}
```

```
(num -> {[x : num] [y : num]})
```

Non-typable avec les règles actuelles :

$$(\text{num} \rightarrow \{[x : \text{num}] [y : \text{num}]\}) \not\leq (\text{num} \rightarrow \{[x : \text{num}]\})$$

Sous-typage et fonctions : Exemple 1

❑ Comparer

$$(\text{num} \rightarrow \{[x : \text{num}] [y : \text{num}]\})$$

et

$$(\text{num} \rightarrow \{[x : \text{num}]\})$$

❑ Règle actuelle

$$(\tau_1 \rightarrow \tau_2) \leq (\tau_1 \rightarrow \tau_2)$$

❑ Nouvelle règle pour autoriser la comparaison

$$\frac{\tau_2 \leq \tau'_2}{(\tau_1 \rightarrow \tau_2) \leq (\tau_1 \rightarrow \tau'_2)}$$

Sous-typage et fonctions : Exemple 2

Ce magasin ne rachète
que des smartphones.

Je veux revendre mon ancien
portable.

$(\{[val : num] \ [gen : num]\} \rightarrow num)$

$(\{[val : num]\} \rightarrow num)$

NON

$(\{[val : num] \ [gen : num]\} \rightarrow num) \not\leq (\{[val : num]\} \rightarrow num)$

Sous-typage et fonctions : Exemple 2

$((\{[x : \text{num}]\} \rightarrow \text{num}) \rightarrow \text{num})$

$\{\{\text{lambda } \{[f : (\{[x : \text{num}]\} \rightarrow \text{num})]\}$
 $\{f \{ \text{record } [x \ 1]\}\}\}$

$\{\text{lambda } \{[r : \{[x : \text{num}] [y : \text{num}]]\}\}$
 $\{\text{get } r \ y\}\}$

$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num})$

Non-typable car

$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num}) \not\leq (\{[x : \text{num}]\} \rightarrow \text{num})$

Sous-typage et fonctions : Exemple 2

❑ Comparer

$(\{[x : \text{num}] \ [y : \text{num}]\} \rightarrow \text{num})$

et

$(\{[x : \text{num}]\} \rightarrow \text{num})$

❑ Règle actuelle

$$\frac{\tau_2 \leq \tau'_2}{(\tau_1 \rightarrow \tau_2) \leq (\tau_1 \rightarrow \tau'_2)}$$

❑ La comparaison est correctement rejetée

Sous-typage et fonctions : Exemple 3

Je connais un autre magasin qui reprend tout type de téléphone.

```
({[val : num]} -> num)
```

Enfinement, le smartphone ne me convient pas...

```
({[val : num] [gen : num]} -> num)
```

OK

$$(\{[val : num]\} \rightarrow num) \leq (\{[val : num] [gen : num]\} \rightarrow num)$$

Sous-typage et fonctions : Exemple 3

```
((([x : num] [y : num]] -> num) -> num)
```

```
{lambda {[f : (([x : num] [y : num]] -> num)]}  
  {f {record [x 1] [y 2]}}}
```

```
{lambda {[r : {[x : num]}]}  
  {get r x}}
```

```
(([x : num]] -> num)
```

Non-typable avec les règles actuelles :

$$([x : num]] -> num) \not\leq ([x : num] [y : num]] -> num)$$

Sous-typage et fonctions : Exemple 3

❑ Comparer

$$(\{[x : \text{num}]\} \rightarrow \text{num})$$

et

$$(\{[x : \text{num}] [y : \text{num}]\} \rightarrow \text{num})$$

❑ Règle actuelle

$$\frac{\tau_2 \leq \tau'_2}{(\tau_1 \rightarrow \tau_2) \leq (\tau_1 \rightarrow \tau'_2)}$$

❑ Nouvelle règle pour autoriser la comparaison

$$\frac{\tau'_1 \leq \tau_1 \quad \tau_2 \leq \tau'_2}{(\tau_1 \rightarrow \tau_2) \leq (\tau'_1 \rightarrow \tau'_2)}$$

Covariance et contravariance

$$\frac{\tau'_1 \leq \tau_1 \quad \tau_2 \leq \tau'_2}{(\tau_1 \rightarrow \tau_2) \leq (\tau'_1 \rightarrow \tau'_2)}$$

- ❑ On dit que le type de retour des fonctions est **covariant** avec le type des fonctions.
- ❑ On dit que le type des paramètres des fonctions est **contravariant** avec le type des fonctions.

Covariance et contravariance

$$\{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\}$$

$$x_i = x'_j \Rightarrow \tau_i \leq \tau'_j$$

$$\{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}$$

□ Le type des champs est covariant avec le type des enregistrements

... du moins tant que les mises à jour sont fonctionnelles.

SOUS-TYPAGE ET MUTATIONS

Sous-typage et mise à jour des champs

$(\{[x : \text{num}]\} \rightarrow \{[x : \text{num}]\})$

$\{\{\text{lambda } [r : \{[x : \text{num}]\}]\}$
 $\{\text{set! } r \ x \ 3\}\}$

$\{\text{record } [x \ 1] \ [y \ 2]\}\}$

$\{[x : \text{num}] \ [y : \text{num}]\}$

$$\frac{\Gamma \vdash \text{expr}_1 : (\tau_1 \rightarrow \tau_2) \quad \Gamma \vdash \text{expr}_2 : \tau_3 \quad \tau_3 \leq \tau_1}{\Gamma \vdash \{\text{expr}_1 \ \text{expr}_2\} : \tau_2}$$

$$\begin{array}{c} \{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\} \\ x_i = x'_j \Rightarrow \tau_i \leq \tau'_j \end{array}$$

$$\{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}$$

Les règles semblent correctes.

Sous-typage et mise à jour des champs

```

{get
  {get
    {{lambda {[r : {[p : {[x : num]}]}]}
      {set! r p {record [x 3]}}}
    {record {p {record [x 1] [y 2]}}}}
  p}
  y}

```

({{[p : {[x : num]}]} -> {[p : {[x : num]}]}})

{[p : {[x : num] [y : num]}]}

Non-typable

Sous-typage et mise à jour des champs

```
{{lambda {[r1 : {[p : {[x : num] [y : num]}]}}}
```

```
{begin
```

```
  {{lambda {[r2 : {[p : {[x : num]}]}}}
```

```
    {set! r2 p {record [x 3]}}
```

```
  r1}
```

```
    {get {get r1 p} y}}
```

```
  {record [p {record [x 1] [y 2]}]}}
```

`{[p : {[x : num]}]` **OK**

`{[p : {[x : num] [y : num]}]}`

OK par sous-typage

`{[p : {[x : num] [y : num]}]}`

OK

`{[p : {[x : num] [y : num]}]}`

OK ou pas...

Sous-typage et mise à jour des champs

```

{{lambda {[r1 : {[p : {[x : num] [y : num]}]}}}
  {begin
    {{lambda {[r2 : {[p : {[x : num]}]}}}
      {set! r2 p {record [x 3]}}}
    r1}
    {get {get r1 p} y}}}
  {record [p {record [x 1] [y 2]}]}}}
  
```

- ❑ L'expression est valide selon le système de typage actuel, mais que se passe-t-il si on tente de l'évaluer ?

Sous-typage et mise à jour des champs

❑ Comparer

$$\{[p : \{[x : \text{num}] [y : \text{num}]]]\}$$

et

$$\{[p : \{[x : \text{num}]]]\}$$

❑ Règle actuelle

$$\frac{\begin{array}{c} \{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\} \\ x_i = x'_j \Rightarrow \tau_i \leq \tau'_j \end{array}}{\{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}}$$

❑ Ne fonctionne pas pour la mise à jour impérative des champs.

Sous-typage et mise à jour des champs

❑ Comparer

$$\{[p : \{[x : \text{num}] [y : \text{num}]]]\}$$

et

$$\{[p : \{[x : \text{num}]]]\}$$

❑ Nouvelle règle

$$\frac{\begin{array}{c} \{x_1, \dots, x_n\} \supseteq \{x'_1, \dots, x'_m\} \\ x_i = x'_j \Rightarrow \tau_i = \tau'_j \end{array}}{\{[x_1 : \tau_1] \dots [x_n : \tau_n]\} \leq \{[x'_1 : \tau'_1] \dots [x'_m : \tau'_m]\}}$$

❑ Avec la mise à jour impérative, le type des champs doit être **invariant**.