Programmation Avancée

Sous-typage: types simples

David Baelde

ENS Paris-Saclay, L3 2020–2021

Le lambda-calcul simplement sous-typé

$$\frac{}{\Gamma, x : \tau \vdash x : \tau} \qquad \frac{}{\Gamma \vdash c : T(c)}$$

$$\frac{\Gamma, x : \tau \vdash u : \tau'}{\Gamma \vdash \lambda x.u : \tau \to \tau'} \qquad \frac{\Gamma \vdash u : \tau \to \tau' \quad \Gamma \vdash v : \tau}{\Gamma \vdash uv : \tau'}$$

Une nouvelle règle utilisant une relation de sous-typage à spécifier :

$$\frac{\Gamma \vdash u : \tau \quad \tau \leq \tau'}{\Gamma \vdash u : \tau'}$$

Intuition

Le type τ est un sous-type de τ' (i.e. $\tau \leq \tau'$) quand :

- Les valeurs de type τ sont aussi des valeurs de type τ'.
- Remplacer une valeur de type τ' par une valeur de type τ ne peut pas provoquer d'erreur à l'exécution.

Exemples : Even $\leq$ Nat, ColoredPoint $\leq$ Point, $\tau \leq$ Any, etc.

Préservation du typage

Théorème

Si $\Gamma \vdash u : \tau$ et $u \leadsto v$ alors $\Gamma \vdash v : \tau$.

Preuve pour le cas où $u = (\lambda x.u_1)u_2 \leadsto u_1\{x \mapsto u_2\}$.

Sans perte de généralité, si $\leq$ est réflexive et transitive, on a :

$$\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \lambda x.u_1 : \tau'' \to \tau'} \qquad \Gamma \vdash u_2 : \tau''}{\Gamma \vdash (\lambda x.u_1)u_2 : \tau'} \qquad \tau' \leq \tau}{\Gamma \vdash (\lambda x.u_1)u_2 : \tau}$$

Préservation du typage

Théorème

Si $\Gamma \vdash u : \tau$ et $u \rightsquigarrow v$ alors $\Gamma \vdash v : \tau$.

Preuve pour le cas où $u = (\lambda x.u_1)u_2 \rightsquigarrow u_1\{x \mapsto u_2\}$.

Sans perte de généralité, si $\leq$ est réflexive et transitive, on a :

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, x : \tau_1 \vdash u_1 : \tau_2}{\Gamma \vdash \lambda x.u_1 : \tau_1 \to \tau_2} \quad \tau_1 \to \tau_2 \leq \tau'' \to \tau'}{\Gamma \vdash \lambda x.u_1 : \tau'' \to \tau'} \qquad \Gamma \vdash u_2 : \tau''}{\Gamma \vdash (\lambda x.u_1)u_2 : \tau'} \qquad \tau' \leq \tau}{\Gamma \vdash (\lambda x.u_1)u_2 : \tau}$$

Préservation du typage

Théorème

Si $\Gamma \vdash u : \tau$ et $u \rightsquigarrow v$ alors $\Gamma \vdash v : \tau$.

Preuve pour le cas où $u = (\lambda x.u_1)u_2 \rightsquigarrow u_1\{x \mapsto u_2\}$.

Sans perte de généralité, si $\leq$ est réflexive et transitive, on a :

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, x : \tau_1 \vdash u_1 : \tau_2}{\Gamma \vdash \lambda x.u_1 : \tau_1 \to \tau_2} \quad {\color{red}\tau_1 \to \tau_2 \leq \tau'' \to \tau'}}{\Gamma \vdash \lambda x.u_1 : \tau'' \to \tau'} \qquad \Gamma \vdash u_2 : \tau''}{\Gamma \vdash (\lambda x.u_1)u_2 : \tau'} \qquad {\color{red}\tau' \leq \tau}}{\Gamma \vdash (\lambda x.u_1)u_2 : \tau}$$

Pour typer $\Gamma \vdash u_1\{x \mapsto u_2\} : \tau$ on a besoin de certaines relations :

- ${\color{red}\tau'' \leq \tau_1}$ pour avoir $\Gamma \vdash u_2 : \tau_1$ et $\Gamma \vdash u_1\{x \mapsto u_2\} : \tau_2$
- ${\color{red}\tau_2 \leq \tau'}$ pour conclure.

Progrès

Théorème

Si u n'est pas une valeur et $\vdash u : \tau$ alors il existe v tel que $u \rightsquigarrow v$.

Preuve.

Les constantes et abstractions sont des valeurs, et u doit être clos.
Il suffit de considérer le cas où $u = u_1 u_2$. On analyse la dérivation de typage :

$$\dfrac{\dfrac{\dfrac{\vdash u_1 : \tau_x \quad \tau_x \leq \tau'' \to \tau'}{\vdash u_1 : \tau'' \to \tau'}}{\vdash u_1 u_2 : \tau'} \qquad \tau' \leq \tau}{\vdash u_1 u_2 : \tau}$$

Progrès

Théorème

Si u n'est pas une valeur et $\vdash u : \tau$ alors il existe v tel que $u \leadsto v$.

Preuve.

Les constantes et abstractions sont des valeurs, et u doit être clos.
Il suffit de considérer le cas où $u = u_1 u_2$. On analyse la dérivation de typage :

$$\dfrac{\dfrac{\dfrac{\vdash u_1 : \tau_x \quad \tau_x \leq \tau'' \to \tau'}{\vdash u_1 : \tau'' \to \tau'}}{\vdash u_1 u_2 : \tau'} \qquad \tau' \leq \tau}{\vdash u_1 u_2 : \tau}$$

On exige que $\tau_x \leq \tau'' \to \tau'$ entraîne que τ_x est une flèche.
Ainsi, u_1 doit être de la forme $(((\lambda x.u_1')v_1)v_2)\ldots v_n)$ et on a un redex.

Sous-typage pour la flèche

Étant donné une relation réflexive et transitive $\leq^B$ sur les types de base T, on peut définir inductivement la relation $\leq$ comme suit :

$$\frac{\mathrm{T} \;\leq^B \mathrm{T'}}{\mathrm{T} \leq \mathrm{T'}} \qquad \frac{\tau_1' \leq \tau_1 \quad \tau_2 \leq \tau_2'}{\tau_1 \to \tau_2 \leq \tau_1' \to \tau_2'}$$

Théorème

La relation $\leq$ est réflexive et transitive.

Elle satisfait les propriétés attendues pour les types $\to$: non-confusion et inversion.

Exemple

Si Even $\leq^B$ Nat, on aura Nat $\to$ Even $\leq$ Even $\to$ Nat.

Autres relations satisfaisant nos contraintes

Moins permissif

Interdire les sous-typages stricts pour les flèches, en remplaçant la règle précédente par celle-ci :

$$\frac{}{\tau \to \tau' \leq \tau \to \tau'}$$

C'est dommage : si $\mathsf{T} \leq^B \mathsf{T}'$ et $f : \mathsf{T}' \to \tau$ on pourra toujours passer un $c : \mathsf{T}$ à f, mais on ne pourra pas passer f à une fonction attendant $\mathsf{T} \to \tau$.

Plus permissif

Ajouter un type Any, et la règle suivante :

$$\frac{}{\tau \leq \mathsf{Any}}$$

On peut voir $f : \tau \to \tau'$ comme $f : \mathsf{Any}$, mais on ne peut pas appliquer $f : \mathsf{Any}$ à un argument.

Variante algorithmique des règles de typage

La règle de sous-typage n'est pas "dirigée par la syntaxe",
elle peut être utilisée n'importe quand, n'importe comment !

On peut s'en passer en annotant les λ et modifiant la règle d'application :

$$\frac{}{\Gamma, x : \tau \vdash_a x : \tau} \qquad \frac{}{\Gamma \vdash_a c : T(c)}$$

$$\frac{\Gamma, x : \tau \vdash_a u : \tau'}{\Gamma \vdash_a \lambda(x : \tau).u : \tau \to \tau'} \qquad \frac{\Gamma \vdash_a u : \tau_1 \to \tau_2 \quad \Gamma \vdash_a v : \tau \quad \tau \leq \tau_1}{\Gamma \vdash_a uv : \tau_2}$$

Ces règles décrivent un algorithme de reconstruction/vérification de types.

Théorème

- Si $\Gamma \vdash_a u : \tau$ alors $\Gamma \vdash u : \tau$.
- Si $\Gamma \vdash u : \tau$ alors il existe $\tau' \leq \tau$ tel que $\Gamma \vdash_a u : \tau'$.