Types raffinements — Refinement types

Examen final du cours MPRI 2-4 / Final exam for MPRI 2-4 course

2013/03/12 — Durée / Duration: 2h45

Le but de l'examen est d'étudier l'extension du λ-calcul simplement typé par des types raffinements. Les sections peuvent être traitées de façon indépendante au sens où les réponses aux questions d'une section ne dépendent pas des réponses aux questions des autres sections.

La syntaxe des termes t et des valeurs v est standard. Les types sont stratifiés en deux classes syntaxiques : les types de base θ et les types structurés τ. Les types structurés contiennent les types produits dépendants et les raffinements de type, de la forme "$\{x : \theta \mid t\}$" qui, informellement, servent à restreindre un type de base θ à l'aide d'un terme booléen t servant de prédicat de filtrage. Dans un type de la forme "$\{x : \theta \mid t\}$", la variable x est liée dans le terme t. Les types raffinements sont donc quotientés par α-conversion des variables liées dans les raffinements. Dans la suite, on notera $\tau \to \tau'$ pour $\Pi(x : \tau).\tau'$ si x n'est pas libre dans τ'.

The goal of this exam is to study an extension of the simply typed λ-calculus with refinement types. The different sections can be treated separately because, for a given section, the answers to the questions of this section do not depend on the answers to the questions of other sections.
The syntax of terms t and values v is standard. The types are stratified into two syntactic classes: basic types θ and structured types τ. The structured types contain dependent product types and refinement types of the form $\{x : \theta \mid t\}$ that, roughly speaking, are used to restrict a basic type θ given a boolean term t that acts as a filtering predicate. In a refinement type $\{x : \theta \mid t\}$, x is bound in the term t. Refinement types of the form $\{x : \theta \mid t\}$ are therefore quotiented with respect to the α-renaming of the variables x. In the sequel, we will write $\tau \to \tau'$ for $\Pi(x : \tau).\tau'$ if x is not free in τ'.

$$t ::= x \mid t\ t \mid \lambda(x : \tau).\, t \mid \mathbf{c} \qquad v ::= \mathbf{c} \mid \lambda(x : \tau).\, t$$

$$\theta ::= \mathbf{int} \mid \mathbf{bool} \mid \mathbf{unit} \qquad \tau ::= \Pi(x : \tau).\tau \mid \{x : \theta \mid t\}$$

On note $\mathbf{c}$ pour désigner une constante prise dans l'ensemble suivant :

We write $\mathbf{c}$ for a constant taken in the following set:

$$\mathbb{B} \cup \{\&, \&_{\mathsf{T}}, \&_{\mathsf{F}}\} \cup \{\mathrm{fix}_\tau, \mathrm{if}_\tau, =_\theta\} \cup \{=_{\theta,b}\}_{b\in\mathbb{B}} \cup \mathbb{O} \cup \{\phi_k\}_{k\in\mathbb{Z}\,\wedge\,\phi\in\mathbb{O}}$$

where $\mathbb{B} = \{()\} \cup \{\mathsf{T}, \mathsf{F}\} \cup \mathbb{Z}$ and $\mathbb{O} = \{>, \geq, -, *\}$.

La sémantique opérationnelle du langage suit une stratégie d'appel par valeur et de réduction faible. On la spécifie à l'aide d'une réduction β et d'une réduction δ qui se réduisent sous un contexte d'évaluation $\mathcal{C}$ dont la syntaxe est :

The operational semantics of the language follows a call-by-value strategy and uses a weak reduction. It is composed of a β-reduction and δ-reduction that are applied under an evaluation context $\mathcal{C}$ whose syntax is:

$$\mathcal{C} ::= v\ [] \mid []\ t$$

Les règles de la sémantique opérationnelle à petits pas sont alors :

Then, the rules for the small-step operational semantics are:

$$\textsc{Red-}\beta\ \frac{}{(\lambda(x:\tau).\,t)\ v \rightsquigarrow t[x \mapsto v]} \qquad \textsc{Red-}\delta\ \frac{}{\mathbf{c}\ v \rightsquigarrow \delta\ (\mathbf{c}, v)} \qquad \textsc{Red-Context}\ \frac{t \rightsquigarrow u}{\mathcal{C}[t] \rightsquigarrow \mathcal{C}[u]}$$

où δ est une fonction partielle d'interprétation des constantes définie ainsi :

$$\begin{array}{rcl|rcl|rcl}
\delta(\&, \mathsf{T}) & = & \&_{\mathsf{T}} & \delta(\mathrm{fix}_\tau, v) & = & v\ (\mathrm{fix}_\tau\ v) & \delta(=_\theta, b) & = & (=_{\theta,b}) \\
\delta(\&, \mathsf{F}) & = & \&_{\mathsf{F}} & \delta(\mathrm{if}_\tau, \mathsf{T}) & = & \lambda(x : \mathbf{unit} \to \tau).\,\lambda(y : \mathbf{unit} \to \tau).\,x\ () & \delta(=_{\theta,b}, b) & = & \mathsf{T} \\
\delta(\&_{\mathsf{T}}, \mathsf{F}) & = & \mathsf{F} & \delta(\mathrm{if}_\tau, \mathsf{F}) & = & \lambda(x : \mathbf{unit} \to \tau).\,\lambda(y : \mathbf{unit} \to \tau).\,y\ () & \delta(=_{\theta,b}, b') & = & \mathsf{F} \quad \text{if } b \neq b' \\
\delta(\&_{\mathsf{T}}, \mathsf{T}) & = & \mathsf{T} & \delta(op, k) & = & op_k & & & \text{where } b, b' \in \mathbb{B} \\
\delta(\&_{\mathsf{F}}, \mathsf{T}) & = & \mathsf{F} & \delta(op_k, k') & = & v \quad \text{where } v = op(k, k') & & & \\
\delta(\&_{\mathsf{F}}, \mathsf{F}) & = & \mathsf{F} & & & & & &
\end{array}$$

Notations On écrira parfois $\Leftrightarrow$ pour $=_{\mathbf{bool}}$ et $=$ pour $=_{\mathbf{int}}$. Pour alléger l'écriture de certains types, on écrira aussi θ au lieu de $\{x : \theta \mid \mathsf{T}\}$ et $\theta\{t\}$ au lieu de $\{x : \theta \mid x =_\theta t\}$.

On se donne maintenant une relation de sous-typage sous un environnement de typage Γ entre deux types τ et τ' dont le jugement s'écrit $\Gamma \vdash \tau \leq \tau'$:

Notations We will sometimes write $\Leftrightarrow$ for $=_{\mathbf{bool}}$ and $=$ for $=_{\mathbf{int}}$. As shorcuts, we will also write θ instead of $\{x : \theta \mid \mathsf{T}\}$ and $\theta\{t\}$ instead of $\{x : \theta \mid x =_\theta t\}$.

We now define a subtyping relation under a typing environment Γ between two types τ and τ' written $\Gamma \vdash \tau \leq \tau'$:

$$\Gamma ::= \emptyset \mid \Gamma, (x : \tau)$$

$$\textsc{Sub-Refl}\ \frac{}{\Gamma \vdash \tau \leq \tau} \qquad \textsc{Sub-Trans}\ \frac{\Gamma \vdash \tau \leq \tau' \qquad \Gamma \vdash \tau' \leq \tau''}{\Gamma \vdash \tau \leq \tau''} \qquad \textsc{Sub-Arrow}\ \frac{\Gamma \vdash \tau_1' \leq \tau_1 \qquad \Gamma, (x : \tau_1') \vdash \tau_2 \leq \tau_2'}{\Gamma \vdash \Pi(x : \tau_1).\tau_2 \leq \Pi(x : \tau_1').\tau_2'} \qquad \textsc{Sub-Set}\ \frac{\Gamma, (x : \theta) \vdash t \Rightarrow u}{\Gamma \vdash \{x : \theta \mid t\} \leq \{x : \theta \mid u\}}$$

Le jugement $\Gamma, (x : \theta) \vdash t \Rightarrow u$ est dérivable si pour toute substitution σ close vis-à-vis de Γ, si $\sigma(t)$ s'évalue en T alors $\sigma(u)$ s'évalue aussi en T. La définition formelle de ce jugement sera l'objet de la section 3.

La relation de sous-typage est injectée dans le système de type à l'aide d'une règle de subsomption.

The judgement $\Gamma, (x : \theta) \vdash t \Rightarrow u$ is derivable if for all substitution σ closed with respect to Γ, if $\sigma(t)$ evaluates into T than $\sigma(u)$ evaluates into T. The formal definition of this judgement will be the topic of the section 3.

The subtyping relation is connected with the type system using a subsumption rule.

$$\textsc{B-Var}\ \frac{(x : \{y : \theta \mid t\}) \in \Gamma}{\Gamma \vdash x : \{y : \theta \mid t\ \&\ y = x\}} \qquad \textsc{P-Var}\ \frac{(x : \Pi(y : \tau_1).\tau_2) \in \Gamma}{\Gamma \vdash x : \Pi(y : \tau_1).\tau_2} \qquad \textsc{Cst}\ \frac{}{\Gamma \vdash \mathbf{c} : \mathrm{ty}(\mathbf{c})}$$

$$\textsc{Lam}\ \frac{\Gamma \vdash \tau_1 \qquad \Gamma, (x : \tau_1) \vdash t : \tau_2}{\Gamma \vdash \lambda(x : \tau_1).\,t : \Pi(x : \tau_1).\tau_2} \qquad \textsc{App}\ \frac{\Gamma \vdash t : \Pi(x : \tau_1).\tau_2 \qquad \Gamma \vdash u : \tau_1}{\Gamma \vdash t\ u : \tau_2[x \mapsto u]}$$

$$\textsc{Sub}\ \frac{\Gamma \vdash t : \tau_1 \qquad \Gamma \vdash \tau_1 \leq \tau_2 \qquad \Gamma \vdash \tau_2}{\Gamma \vdash t : \tau_2} \qquad \frac{\Gamma \vdash \tau_1 \qquad \Gamma, (x : \tau_1) \vdash \tau_2}{\Gamma \vdash \Pi(x : \tau_1).\tau_2} \qquad \frac{\Gamma, (x : \theta) \vdash t : \mathbf{bool}}{\Gamma \vdash \{x : \theta \mid t\}}$$

On supposera que les environnements de typage Γ contiennent des variables toutes distinctes. Ainsi, on les utilisera librement comme des fonctions à support fini dont on notera "$\Gamma\,(x)$" l'évaluation en la variable x lorsque x est dans le support de Γ.

We will assume that typing environments Γ contain distinct variables. That way, they will be freely used as functions of finite support. We will write "$\Gamma\,(x)$" to look for the image of the variable x through Γ, if x is in the support of Γ.

La fonction ty(**c**) associe un type à chaque constante **c** :

The ty(**c**) maps a constant to its type.

$$\begin{array}{rcll}
\mathrm{ty}(()) & = & \mathbf{unit} & \\
\mathrm{ty}(\mathsf{T}) & = & \mathbf{bool}\{\mathsf{T}\} & \\
\mathrm{ty}(\mathsf{F}) & = & \mathbf{bool}\{\mathsf{F}\} & \\
\mathrm{ty}(n) & = & \mathbf{int}\{n\} & \text{where } n \in \mathbb{Z} \\
\mathrm{ty}(\mathrm{fix}_\tau) & = & (\tau \to \tau) \to \tau & \\
\mathrm{ty}(\mathrm{if}_\tau) & = & \mathbf{bool} \to (\mathbf{unit} \to \tau) \to (\mathbf{unit} \to \tau) \to \tau & \\
\mathrm{ty}(op) & = & \Pi(y : \mathrm{ity}(op)).\Pi(x : \mathrm{ity}(op)).\{z : \mathrm{oty}(op) \mid z =_{\mathrm{oty}(op)} (y\, op\, x)\} & \\
\mathrm{ty}(op_b) & = & \Pi(x : \mathrm{ity}(op)).\{z : \mathrm{oty}(op) \mid z =_{\mathrm{oty}(op)} (b\, op\, x)\} &
\end{array}$$

$$\text{where } \mathrm{ity}(op) = \begin{cases} \mathbf{int} & \text{if } op \in \{\geq, >, *, -\} \\ \mathbf{bool} & \text{if } op \in \{\wedge\} \\ \theta & \text{if } op \in \{=_\theta\} \end{cases} \quad \text{and } \mathrm{oty}(op) = \begin{cases} \mathbf{int} & \text{if } op \in \{*, -\} \\ \mathbf{bool} & \text{if } op \in \{\geq, >, =_\theta, \wedge\} \end{cases}$$

1 Échauffement / Warm up

Question 1

À l'aide des types déclarés plus haut, donnez une dérivation de typage pour le jugement suivant :

Based on the earlier type declarations, give a typing derivation for the following judgement:

$$\emptyset \vdash \lambda(x : \mathbf{int}).\,(x * x) : \Pi(x : \mathbf{int}).\{n : \mathbf{int} \mid n = x * x\}$$

On ne justifiera pas les applications des règles (Sub-Set).

The applications of the rule (Sub-Set) are not to be justified.

Answer: Let us notice, once and for the rest of this exam, that for all Γ, θ and x, $\Gamma \vdash \theta\{x\} \leq \theta$ holds. Indeed, we have:

$$\text{Sub-Set}\ \frac{\Gamma, (y : \theta) \vdash y = t \Rightarrow \mathsf{T}}{\Gamma \vdash \theta\{t\} \leq \{y : \theta \mid \mathsf{T}\}}$$

We will freely invoke this derivable axiom as (Sub-Basic) in the sequel. Similarly, let us notice that for all Γ and θ, the judgment $\Gamma \vdash \theta$ holds. We will invoke this derivable axiom (Wf-Basic) in the sequel.

Finally, let us notice that the following rule (Basic-BVar) is derivable in the system:

$$\text{Basic-B-Var}\ \frac{(x : \theta) \in \Gamma}{\Gamma \vdash x : \theta}$$

Indeed, we have:

$$\text{Sub}\ \frac{\text{B-Var}\ \dfrac{(x : \theta) \in \Gamma}{\Gamma \vdash x : \{y : \theta \mid \mathsf{T}\,\&\,y = x\}} \qquad \dfrac{}{\Gamma \vdash \{y : \theta \mid \mathsf{T}\,\&\,y = x\} \leq \theta}\ \text{Sub-Basic}}{\Gamma \vdash x : \theta}$$

Then a typing derivation for the judgement is:

$$\text{Lam}\ \frac{\text{App}\ \dfrac{\text{App}\ \dfrac{\text{Cst}\ \dfrac{}{\Gamma \vdash * : \Pi(z : \boldsymbol{int}).\Pi(y : \boldsymbol{int}).\{n : \boldsymbol{int} \mid n = z * y\}} \quad \dfrac{\text{Basic-BVar}\ \dfrac{(x : \boldsymbol{int}) \in \Gamma}{\Gamma \vdash x : \boldsymbol{int}}}{\Gamma \vdash *x : \Pi(y : \boldsymbol{int}).\{n : \boldsymbol{int} \mid n = x * y\}} \quad \dfrac{\text{Basic-BVar}\ \dfrac{(x : \boldsymbol{int}) \in \Gamma}{\Gamma \vdash x : \boldsymbol{int}}}{\Gamma \vdash x * x : \{n : \boldsymbol{int} \mid n = x * x\}} \quad \dfrac{}{\emptyset \vdash \boldsymbol{int}}\ \text{Wf-Basic}}{\emptyset \vdash \lambda(x : \boldsymbol{int}).\,(x * x) : \Pi(x : \boldsymbol{int}).\{n : \boldsymbol{int} \mid n = x * x\}}$$

where $\Gamma \equiv (x : \textbf{\textit{int}})$.

Question 2

En déduire une dérivation de typage pour le jugement suivant :

Deduce from the last answer a typing derivation for the following judgement:

$$\emptyset \vdash \lambda(x : \textbf{int}).\,(x * x) : \textbf{int} \to \textbf{pos}$$

avec $\textbf{pos} \equiv \{k : \textbf{int} \mid k \geq 0\}$. On ne justifiera pas les applications des règles (Sub-Set).

where $\textbf{pos} \equiv \{k : \textbf{int} \mid k \geq 0\}$. The applications of the rule (Sub-Set) are not to be justified.

Answer:

Let us write $\mathcal{D}_{Q1}$ *for the answer to the previous question. Then, a typing derivation for:*

$$\emptyset \vdash \lambda(x : \textbf{\textit{int}}).\,(x * x) : \textbf{\textit{int}} \to \textbf{\textit{pos}}$$

is:

$$\dfrac{\mathcal{D}_{Q1} \qquad \dfrac{\text{Sub-Refl}\ \dfrac{}{\emptyset \vdash \textbf{\textit{int}} \leq \textbf{\textit{int}}} \qquad \dfrac{\dfrac{}{(x : \textbf{\textit{int}}), (n : \textbf{\textit{int}}) \vdash n = x * x \Rightarrow n \geq 0}\ \text{Sub-Set}}{(x : \textbf{\textit{int}}) \vdash \{n : \textbf{\textit{int}} \mid n = x * x\} \leq \{n : \textbf{\textit{int}} \mid n \geq 0\}}\ \text{Sub-Arrow}}{\emptyset \vdash \Pi(x : \textbf{\textit{int}}).\{n : \textbf{\textit{int}} \mid n = x * x\} \leq \textbf{\textit{int}} \to \textbf{\textit{pos}}}\ \text{Sub}}{\emptyset \vdash \lambda(x : \textbf{\textit{int}}).\,(x * x) : \textbf{\textit{int}} \to \textbf{\textit{pos}}}$$

with $\mathcal{D}_1$ *equal to:*

$$\dfrac{\text{Wf-Basic}\ \dfrac{}{\emptyset \vdash \textbf{\textit{int}}} \qquad \dfrac{\dfrac{\mathcal{D}_2 \qquad \dfrac{\text{Sub-Basic}\ \dfrac{}{\Gamma \vdash \{b : \textbf{\textit{bool}} \mid b = y \geq 0\} \leq \textbf{\textit{bool}}} \qquad \text{Wf-Basic}\ \dfrac{}{\Gamma \vdash \textbf{\textit{bool}}}}{\Gamma \vdash y \geq 0 : \textbf{\textit{bool}}}\ \text{Sub}}{(x : \textbf{\textit{int}}) \vdash \textbf{\textit{pos}}}\ \text{Wf-Set}}{\emptyset \vdash \textbf{\textit{int}} \to \textbf{\textit{pos}}}\ \text{Wf-Arrow}$$

with $\mathcal{D}_2$ *is*

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \geq : ty(\geq)}\ \text{Cst} \qquad \dfrac{}{\Gamma \vdash y : \textbf{\textit{int}}}\ \begin{array}{c}\text{Basic-B-Var}\\ (y : \textbf{\textit{int}}) \in \Gamma\end{array}}{\Gamma \vdash (\geq\ y) : \Pi(x : \textbf{\textit{int}}).\{b : \textbf{\textit{bool}} \mid b = y \geq x\}}\ \text{App} \qquad \dfrac{\dfrac{}{\Gamma \vdash 0 : \textbf{\textit{int}}\{0\}}\ \text{Cst} \qquad \dfrac{}{\Gamma \vdash \textbf{\textit{int}}\{0\} \leq \textbf{\textit{int}}}\ \text{Sub-Basic} \qquad \dfrac{}{\Gamma \vdash \textbf{\textit{int}}}\ \text{Wf-Basic}}{\Gamma \vdash 0 : \textbf{\textit{int}}}\ \text{Sub}}{\Gamma \vdash y \geq 0 : \{b : \textbf{\textit{bool}} \mid b = y \geq 0\}}\ \text{App}$$

where $\Gamma \equiv (x : \textbf{\textit{int}}), (y : \textbf{\textit{int}})$ *and* $\Gamma' \equiv \Gamma, (b : \textbf{\textit{bool}})$.

Question 3

Expliquez informellement pourquoi le type de la constante "if_τ" n'est pas suffisamment précis pour affecter le type τ_{abs} suivant :

Explain informally why the type of the constant "if_τ" is not precise enough to assign the following type τ_{abs}:

$$\Pi(n : \textbf{int}).\{k : \textbf{int} \mid k \geq 0\}$$

à la fonction suivante :

to the following function:

$$\lambda(x : \textbf{int}).\,\text{if}_{\textbf{pos}}\ (x \geq 0)\ (\lambda(u : \textbf{unit}).\,x)\ (\lambda(u : \textbf{unit}).\,0 - x)$$

Answer: At some point, the typing judgement $(x : \textbf{\textit{int}}), (u : \textbf{\textit{unit}}) \vdash x : \textbf{\textit{pos}}$ *is required but it cannot be derived because there exist integers that are not positive. The problem is rooted in the type of if that does not convey a static information about the dynamic behavior of the boolean expression* $x \geq 0$.

Question 4

On redéfinit la constante if_τ ainsi :

$$\begin{array}{rcl} \text{ty}(\text{if}_\tau) & = & \Pi(x : \textbf{bool}).(\{u : \textbf{unit} \mid x \Leftrightarrow \mathsf{T}\} \to \tau) \to (\{u : \textbf{unit} \mid x \Leftrightarrow \mathsf{F}\} \to \tau) \to \tau \\ \delta(\text{if}_\tau, \mathsf{T}) & = & \lambda(x : \{\text{u} : \textbf{unit} \mid \mathsf{T} \Leftrightarrow \mathsf{T}\} \to \tau).\, \lambda(y : \{\text{u} : \textbf{unit} \mid \mathsf{T} \Leftrightarrow \mathsf{F}\} \to \tau).\, x\ () \\ \delta(\text{if}_\tau, \mathsf{F}) & = & \lambda(x : \{\text{u} : \textbf{unit} \mid \mathsf{F} \Leftrightarrow \mathsf{T}\} \to \tau).\, \lambda(y : \{\text{u} : \textbf{unit} \mid \mathsf{F} \Leftrightarrow \mathsf{F}\} \to \tau).\, y\ () \end{array}$$

Donnez une dérivation de typage pour affecter le type τ_{abs} au terme de la question précédente. On ne justifiera pas les applications des règles (Sub-Set).

Give a typing derivation to assign the type τ_{abs} to the term of the previous question. The applications of the rule (Sub-Set) are not to be justified.

Answer:

$$\textsc{Lam}\ \frac{\textsc{App}\ \dfrac{\textsc{App}\ \dfrac{\textsc{Cst}\ \dfrac{}{(x : \textbf{int}) \vdash \mathit{if}_{pos} : \mathit{ty}(\mathit{if}_{pos})} \quad \mathcal{D}_1}{(x : \textbf{int}) \vdash \mathit{if}_{pos}\ (x \geq 0)\ (\lambda(u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{T}\}).\, x) : (\{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{F}\} \to \textbf{pos}) \to \textbf{pos}} \quad \mathcal{D}_2}{(x : \textbf{int}) \vdash \mathit{if}_{pos}\ (x \geq 0)\ (\lambda(u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{T}\}).\, x)\ (\lambda(u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{F}\}).\, 0 - x) : \textbf{pos}} \quad \frac{\textsc{Wf-Basic}}{\Gamma \vdash \textbf{int}}}{\emptyset \vdash \lambda(x : \textbf{int}).\, \mathit{if}_{pos}\ (x \geq 0)\ (\lambda(u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{T}\}).\, x)\ (\lambda(u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{F}\}).\, 0 - x) : \tau_{abs}}$$

where $\mathcal{D}_1$ is:

$$\textsc{Lam}\ \frac{\textsc{Sub}\ \dfrac{\textsc{B-Var}\ \dfrac{}{\Gamma \vdash x : \textbf{int}\{x\}} \quad \textsc{Sub-Set}\ \dfrac{\textit{(Assumed)}}{\Gamma \vdash \textbf{int}\{x\} \leq \textbf{pos}} \quad \dfrac{\textit{(Question 2)}}{\Gamma \vdash \textbf{pos}}}{\Gamma \vdash x : \textbf{pos}}}{(x : \textbf{int}) \vdash \lambda(u : \{u \Leftrightarrow \mathsf{T} : \textbf{unit} \mid x \geq 0\}).\, x : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{T}\} \to \textbf{pos}}$$

where $\Gamma \equiv (x : \textbf{int}), (u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{T}\})$.

*Thanks to the dependent type of if, the typing environment under which each branch is typed contains an extra hypothesis about the value of the boolean expression. The previous judgement $(x : \textbf{int}), (u : \textbf{unit}) \vdash x : \textbf{pos}$ is now replaced with $(x : \textbf{int}), (u : \{u : \textbf{unit} \mid x \geq 0 \Leftrightarrow \mathsf{T}\}) \vdash x : \textbf{pos}$ which is derivable. Indeed, this judgement can be obtained by an application of (Sub) on $(x : \textbf{int}), (u : \{u : \textbf{unit} \mid x \geq 0 \Leftrightarrow \mathsf{T}\}) \vdash x : \textbf{int}\{x\}$ and $(x : \textbf{int}), (u : \{u : \textbf{unit} \mid x \geq 0 \Leftrightarrow \mathsf{T}\}) \vdash \{z : \textbf{int} \mid z = x\} \leq \{z : \textbf{int} \mid z \geq 0\}$ **(1)**. The judgement (1) is derivable from:*

$$(x : \textbf{int}), (u : \{u : \textbf{unit} \mid x \geq 0 \Leftrightarrow \mathsf{T}\}), (z : \textbf{int}) \vdash z = x \Rightarrow z \geq 0$$

A closed substitution σ that is compatible with this typing environment must assign a positive integer to x for the refinement $\sigma(x \geq 0)$ to reduce to T. Besides if $\sigma(z = x)$ reduces to T then $\sigma(z) = \sigma(x) \geq 0$ which implies that $\sigma(z \geq 0)$ reduces to T.

$\mathcal{D}_2$ is:

$$\textsc{Lam}\ \frac{\textsc{Sub}\ \dfrac{\mathcal{D}_3 \quad \textsc{Sub-Set}\ \dfrac{\textit{(Assumed)}}{\Gamma \vdash \textbf{int}\{0 - x\} \leq \textbf{pos}} \quad \textsc{Wf-SubSet}\ \dfrac{\textit{(See question 2)}}{\Gamma \vdash \textbf{pos}}}{\Gamma \vdash (0 - x) : \textbf{pos}}}{(x : \textbf{int}) \vdash \lambda(u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{F}\}).\, (0 - x) : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{F}\} \to \textbf{pos}}$$

where $\Gamma \equiv (x : \textbf{int}), (u : \{u : \textbf{unit} \mid (x \geq 0) \Leftrightarrow \mathsf{F}\})$ and $\mathcal{D}_3$ is:

$$\textsc{App}\ \frac{\textsc{Cst}\ \dfrac{}{\Gamma \vdash (-) : \mathit{ty}(-)} \quad \textsc{Sub}\ \dfrac{\textsc{Cst}\ \dfrac{}{\Gamma \vdash 0 : \textbf{int}\{0\}} \quad \dfrac{}{\Gamma \vdash \textbf{int}\{0\} \leq \textbf{int}}\ \textsc{Sub-Basic}}{\Gamma \vdash 0 : \textbf{int}}}{\Gamma \vdash (-)\ 0 : \Pi y.\{z : \textbf{int} \mid z = 0 - y\}} \quad \textsc{App}\ \frac{\textsc{Basic-B-Var}\ \dfrac{(x : \textbf{int}) \in \Gamma}{\Gamma \vdash x : \textbf{int}}}{\Gamma \vdash (0 - x) : \textbf{int}\{0 - x\}}$$

The same kind of arguments as for $\mathcal{D}_1$ justifies the validity of the application of (Sub-Set).

2 Sûreté du typage / Type Safety

Question 5

Définissez une fonction $\lfloor \bullet \rfloor$ d'effacement des raffinements dans les types qui associe à chaque type τ, un type simple de la forme :

Define an erasure function $\lfloor \bullet \rfloor$ of type refinements that maps to every type τ, a simple type of the form:

$$\rho ::= \theta \mid \rho \to \rho$$

Answer:

$$\begin{array}{rcl} \lfloor \{x : \theta \mid t\} \rfloor & = & \theta \\ \lfloor \Pi(x : \tau_1).\tau_2 \rfloor & = & \lfloor \tau_1 \rfloor \to \lfloor \tau_2 \rfloor \end{array}$$

On étend la fonction $\lfloor \bullet \rfloor$ aux termes et aux environnements de typage. De plus, on définit une fonction $\mathrm{ty}_{\mathsf{s}}(\mathbf{c}) = \lfloor \mathrm{ty}(\mathbf{c}) \rfloor$ qui doit être utilisée par la règle de typage simple des constantes :

We extend the function $\lfloor \bullet \rfloor$ to terms and typing environments. Besides, we define $\mathrm{ty}_{\mathsf{s}}(\mathbf{c}) = \lfloor \mathrm{ty}(\mathbf{c}) \rfloor$ to be used in the typing rule for constants:

$$\text{S-Cst}\ \frac{}{\Gamma \vdash_{\mathsf{s}} \mathbf{c} : \mathrm{ty}_{\mathsf{s}}(\mathbf{c})}$$

Les autres règles du jugement $\Gamma \vdash_{\mathsf{s}} t : \rho$ sont les règles de typage standard du λ-calcul simplement typé.

The other typing rules defining the judgement $\Gamma \vdash_{\mathsf{s}} t : \rho$ are the standard typing rules for the simply typed λ-calculus.

Question 6

Prouvez le lemme suivant :

Prove the following lemma:

Lemma 1 *If $\Gamma \vdash \tau_1 \leq \tau_2$ holds then $\lfloor \tau_1 \rfloor = \lfloor \tau_2 \rfloor$.*

Answer:By induction over subtyping derivations. The cases (Sub-Refl) and (Sub-Trans) are implied by the reflexivity and transitivity of the equality on the induction hypotheses. The case (Sub-Arrow) follows immediately from the induction hypotheses. In the case (Sub-Set), the subtyping judgement is $\Gamma \vdash \{x : \theta \mid t\} \leq \{x : \theta \mid u\}$ and we must prove that $\lfloor \{x : \theta \mid t\} \rfloor$ is equal to $\lfloor \{x : \theta \mid u\} \rfloor$; this obviously holds since both terms are equal to θ.

Question 7

Prouvez que si $\Gamma \vdash t : \tau$ alors son effacement est bien typé dans le λ-calcul simplement typé, *i.e.* $\lfloor \Gamma \rfloor \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau \rfloor$.

Prove that if $\Gamma \vdash t : \tau$ holds, then its erasure is well-typed in the simply-typed λ-calculus, *i.e.* $\lfloor \Gamma \rfloor \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau \rfloor$ holds.

Answer:By induction over typing derivations.

- *Case (B-Var) and (P-Var): From the hypothesis $(x : \tau) \in \Gamma$, we have $(x : \lfloor \tau \rfloor) \in \lfloor \Gamma \rfloor$. Therefore, $\lfloor \Gamma \rfloor \vdash_{\mathsf{s}} x : \lfloor \tau \rfloor$ by the rule for variables.*
- *Case (Cst): We must show $\lfloor \Gamma \rfloor \vdash_{\mathsf{s}} c : \lfloor ty(c) \rfloor$,* i.e. *$\lfloor \Gamma \rfloor \vdash_{\mathsf{s}} c : ty_{\mathsf{s}}(c)$, which holds by the rule for constants.*
- *Case (Lam): By inversion of the rule, we have $\Gamma, (x : \tau_1) \vdash t : \tau_2$. By induction hypothesis, this judgement leads to: $\lfloor \Gamma, (x : \tau_1) \rfloor \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau_2 \rfloor$, that is: $\lfloor \Gamma \rfloor, (x : \lfloor \tau_1 \rfloor) \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau_2 \rfloor$. The conclusion $\lfloor \Gamma \rfloor \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau_1 \rfloor \to \lfloor \tau_2 \rfloor$ follows by the rule for λ-abstractions.*

- *Case* (App)*: By inversion, we have* $\Gamma \vdash t : \Pi(x : \tau_1).\tau_2$ *and* $\Gamma \vdash u : \tau_1$*. By induction, we get* $\lfloor\Gamma\rfloor \vdash_s \lfloor t\rfloor : \lfloor\Pi(x : \tau_1).\tau_2\rfloor$ ***(1)*** *and* $\lfloor\Gamma\rfloor \vdash_s \lfloor u\rfloor : \lfloor\tau_1\rfloor$ ***(2)****. (1) rewrites into* $\lfloor\Gamma\rfloor \vdash_s \lfloor t\rfloor : \lfloor\tau_1\rfloor \to \lfloor\tau_2\rfloor$ ***(3)****. By the rule for application applied to (2) and (3), we have* $\lfloor\Gamma\rfloor \vdash_s t\ u : \lfloor\tau_2\rfloor$*. It remains to show that* $\lfloor\tau_2\rfloor \equiv \lfloor\tau_2[x \mapsto u]\rfloor$*. The following lemma is necessary:*

  Lemma 2 *For all type* τ *x and t,* $\lfloor\tau\rfloor \equiv \lfloor\tau[x \mapsto t]\rfloor$*. Proof. By induction over types.*

  - *Case* $\tau \equiv \{y : \theta \mid u\}$*: Without loss of generality, we can assume that* $y \neq x$*. In that case,* $\lfloor\{y : \theta \mid u\}\rfloor = \theta$ *and* $\lfloor\{y : \theta \mid u\}[x \mapsto t]\rfloor = \lfloor\{y : \theta \mid u[x \mapsto t]\}\rfloor = \theta$*.*
  - *Case* $\tau \equiv \Pi(y : \tau_1).\tau_2$*: Without loss of generality, we can assume that* $y \neq x$*.* $\lfloor\Pi(y : \tau_1).\tau_2\rfloor = \lfloor\tau_1\rfloor \to \lfloor\tau_2\rfloor$ *and* $\lfloor\Pi(y : \tau_1).\tau_2[x \mapsto u]\rfloor = \lfloor\Pi(y : \tau_1[x \mapsto u]).(\tau_2[x \mapsto u])\rfloor = \lfloor\tau_1[x \mapsto u]\rfloor \to \lfloor\tau_2[x \mapsto u]\rfloor$*. By induction,* $\lfloor\tau_1[x \mapsto u]\rfloor = \lfloor\tau_1\rfloor$ *and* $\lfloor\tau_2[x \mapsto u]\rfloor = \lfloor\tau_2\rfloor$*.*

  Qed.

- *Case* (Sub)*: By inversion, we have* $\Gamma \vdash t : \tau_1$ ***(1)*** *and* $\Gamma \vdash \tau_1 \leq \tau_2$*. By Lemma 1,* $\lfloor\tau_1\rfloor = \lfloor\tau_2\rfloor$*. Thus, by application of the induction hypothesis on (1), we get the expected result.*

Question 8

On admet que le système de type $\vdash_s$ est sûr. Montrez que le système de type avec types raffinements est sûr : si $\vdash t : \tau$ alors le terme t s'évalue sans erreur.

We admit that the type system $\vdash_s$ is sound. Show that the type system with refinement types is sound: if $\vdash t : \tau$ holds then the term t does not go wrong.

Answer:If $\vdash t : \tau$*, then* $\vdash_s \lfloor t\rfloor : \lfloor\tau\rfloor$*. By the soundness of* $\vdash_s$*, the evaluation of* $\lfloor t\rfloor$ *does not go wrong, so does the term t that follows exactly the same reduction steps.*

Question 9

Quelle garantie supplémentaire apportent les types raffinements en sus de la sûreté de l'évaluation ?

What extra guarantee do the refinement types provide in addition to the safety of the evaluation?

Answer:If a term of type $\{x : \theta \mid t\}$ *converges to a value v, this value fullfills the contract of its refinement types, that is* $t[x \mapsto v] \rightsquigarrow^\star \mathsf{T}$*.*

3 Typage sémantique / Semantic typing

On revient maintenant sur la définition formelle du jugement $\Gamma \vdash \tau_1 \leq \tau_2$. En particulier, la règle :

We now come back to the formal definition of the judgement $\Gamma \vdash \tau_1 \leq \tau_2$. In particular, the rule:

$$\textsc{Sub-Set}\quad \frac{\Gamma, (x : \theta) \vdash t \Rightarrow u}{\Gamma \vdash \{x : \theta \mid t\} \leq \{x : \theta \mid u\}}$$

fait référence au jugement "$\Gamma, (x : \theta) \vdash t \Rightarrow u$" que l'on souhaiterait définir de la façon suivante :

refers to the judgement "$\Gamma, (x : \theta) \vdash t \Rightarrow u$" that we would like to define in the following way:

$$\textsc{Implication}\quad \frac{\forall \sigma, \text{if } \vdash \sigma : \Gamma \text{ and } \sigma(t) \rightsquigarrow^\star \mathsf{T} \text{ then } \sigma(u) \rightsquigarrow^\star \mathsf{T}}{\Gamma, (x : \theta) \vdash t \Rightarrow u}$$

où :

where:

$$\textsc{Closed-Substitution}\quad \frac{\forall x \in \mathrm{dom}(\sigma), \quad \vdash \sigma(x) : \Gamma(x)}{\vdash \sigma : \Gamma}$$

Cependant, rajouter ces règles mutuellement récursives dans le système précédent pose problème. En effet, la récursion mutuelle mise en jeu est mal fondée car le jugement de typage apparaît en position négative dans la règle (Implication) (ce qui rend impossible la preuve par induction mutuelle sur les dérivations de ces jugements).

Pour briser cette circularité, on définit une notion d'interprétation d'un type τ (par induction sur $\lfloor\tau\rfloor$) en des ensembles $[\![\tau]\!]$ de termes clos t tels que $\lfloor t\rfloor$ est de type $\lfloor\tau\rfloor$ et vérifie le contrat représenté par son raffinement (si il converge). On remplace alors la règle (Closed-Substitution) par la règle suivante :

Yet, adding these mutually recursive inference rules in the previous system is not possible. Indeed, this mutual recursion is ill-founded because the typing judgement appears in negative position in the rule (Implication) (which prevents proving by mutual induction over the proofs for these judgements).

To break this circularity, we define a notion of interpretation of a type τ (by induction over $\lfloor\tau\rfloor$) as sets of closed terms t such that $\lfloor t\rfloor$ is of type $\lfloor\tau\rfloor$ and verifies the contract defined by its refinement. We replace the rule (Closed-Substitution) by the following rule:

$$\textsc{Semantically-Closed-Substitution}\quad \frac{\forall x \in \mathrm{dom}(\sigma), \quad \sigma(x) \in [\![\sigma(\Gamma(x))]\!]}{\vdash \sigma : \Gamma}$$

Les notions de sous-typage sémantique et de typage associées sont alors :

Then, the related notions of semantic subtyping and typing are:

$$\textsc{Semantic-Typing}\quad \frac{\forall \sigma, \quad \text{if } \vdash \sigma : \Gamma \text{ then } \sigma(t) \in [\![\sigma(\tau)]\!]}{\Gamma \vdash t \in \tau}$$

$$\textsc{Semantic-Subtyping}\quad \frac{\forall \sigma, \quad \text{if } \vdash \sigma : \Gamma \text{ then } [\![\sigma(\tau)]\!] \subseteq [\![\sigma(\tau)']\!]}{\Gamma \vdash \tau \subseteq \tau'}$$

Question 10

Montrez que :

Prove that:

Lemma 3 *If* $\Gamma \vdash \tau \subseteq \tau'$ *and* $\Gamma \vdash t \in \tau$ *then* $\Gamma \vdash t \in \tau'$

Answer: By inversion of $\Gamma \vdash \tau \subseteq \tau'$ *(1), we have* $\forall \sigma$, *if* $\vdash \sigma : \Gamma$ *then* $[\![\sigma(\tau)]\!] \subseteq [\![\sigma(\tau')]\!]$ *(2). By inversion of* $\Gamma \vdash t \in \tau$ *(3), we have* $\forall \sigma$, *if* $\vdash \sigma : \Gamma$ *then* $\sigma(t) \in [\![\sigma(\tau)]\!]$ *(4). Let us take* σ *such that* $\vdash \sigma : \Gamma$. *By (4),* $\sigma(t) \in [\![\sigma(\tau)]\!]$ *and by (2),* $\sigma(t) \in [\![\sigma(\tau)']\!]$, *which is exactly what is needed to apply* (Semantic-Typing) *to conclude.*

Ces notions doivent être compatibles avec les relations de sous-typage et typage syntaxiques :

These notions must be compatible with the syntactic subtyping and typing relations:

Lemma 4

(1) If $\Gamma \vdash \tau \leq \tau'$ *holds, then* $\Gamma \vdash \tau \subseteq \tau'$ *holds.*

(2) If $\Gamma \vdash t : \tau$ *holds, then* $\Gamma \vdash t \in \tau$ *holds.*

Ce lemme va être démontré dans les questions suivantes.

This lemma will be proved in the next questions.

Question 11

Donnez un exemple de terme clos qui converge et un exemple de terme clos qui diverge, tous les deux habitants de $[\![\{x : \mathbf{int} \mid x \geq 0\}]\!]$.

Give an example of a closed term that converges and an example of a closed term that diverges, such that both of them are inhabitants of $[\![\{x : \mathbf{int} \mid x \geq 0\}]\!]$.

Answer: Take 0 for the term that converges and $\mathit{fix}_{\mathbf{int}}\ (\lambda(x : \mathbf{int}).\ x)$ *for the term that diverges.*

Voici une définition incorrecte de $[\![\bullet]\!]$:

Here is an incorrect definition of $[\![\bullet]\!]$:

$$\begin{array}{rcl} [\![\{x : \theta \mid u\}]\!] & = & \{\, t \mid \vdash_{\mathsf{s}} \lfloor t \rfloor : \theta \wedge (u[x \mapsto t] \rightsquigarrow^{\star} \mathsf{T}) \,\} \\ [\![\Pi(x : \tau_1).\tau_2]\!] & = & \{\, t \mid \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau_1 \rfloor \to \lfloor \tau_2 \rfloor \wedge (\forall u \in [\![\tau_1]\!], t\ u \in [\![\tau_2[x \mapsto u]]\!]) \,\} \end{array}$$

En effet, avec cette définition, les termes divergents ne sont pas nécessairement des habitants de l'interprétation de leur type.

Indeed, with that definition, divergent terms may not inhabit the interpretation of their types.

Question 12

Proposez une version corrigée de la définition de $[\![\bullet]\!]$ et vérifiez que vos deux exemples de la question précédente sont validés par cette définition.

Give a fixed version of the definition of $[\![\bullet]\!]$ and check that your two examples of the previous question are validated by this definition.

Answer:

$$\begin{array}{rcl} [\![\{x : \theta \mid u\}]\!] & = & \{\, t \mid \vdash_{\mathsf{s}} \lfloor t \rfloor : \theta \wedge (\lfloor t \rfloor \rightsquigarrow^{\star} \boldsymbol{c} \textit{ implies } u[x \mapsto \boldsymbol{c}] \rightsquigarrow^{\star} \mathsf{T}) \,\} \\ [\![\Pi(x : \tau_1).\tau_2]\!] & = & \{\, t \mid \vdash_{\mathsf{s}} \lfloor t \rfloor : \lfloor \tau_1 \rfloor \to \lfloor \tau_2 \rfloor \wedge (\forall u \in [\![\tau_1]\!], t\ u \in [\![\tau_2[x \mapsto u]]\!]) \,\} \end{array}$$

Question 13

Prouvez que votre définition de $[\![\bullet]\!]$ vérifie la première propriété du Lemme 4.

Prove that your definition of $[\![\bullet]\!]$ verifies the first property of Lemma 4.

Answer: The proof is by induction over the subtyping derivation of $\Gamma \vdash \tau \leq \tau'$. *The cases* (Sub-Refl) *and* (Sub-Trans) *follow from the reflexivity and transitivity of the inclusion relation on sets.*

- *Case* (Sub-Arrow)*: The subtyping derivation ends with* $\Gamma \vdash \Pi(x : \tau_1).\tau_2 \leq \Pi(x : \tau_1').\tau_2'$. *The premises are* $\Gamma \vdash \tau_1' \leq \tau_1$ ***(1)*** *and* $\Gamma, (x : \tau_1') \vdash \tau_2 \leq \tau_2'$ ***(2)***. *By induction, we have* $\Gamma \vdash \tau_1' \subseteq \tau_1$ ***(3)*** *and* $\Gamma, (x : \tau_1) \vdash \tau_2 \subseteq \tau_2'$ ***(4)***. *Let* σ *be such that* $\vdash \sigma : \Gamma$ *and* $t \in [\![\sigma(\Pi(x : \tau_1).\tau_2)]\!]$, i.e. $t \in [\![\Pi(x : \sigma(\tau_1)).\sigma(\tau_2)]\!]$. *Then, for all* $u \in [\![\sigma(\tau_1')]\!]$, *(3) gives* $u \in [\![\sigma(\tau_1)]\!]$ ***(5)***. *Therefore,* $t\ u \in [\![(\sigma + x \mapsto u)(\tau_2)]\!]$. *We also have* $\vdash (\sigma + x \mapsto u) : \Gamma, (x : \tau_1)$ *because of (5). Hence,* $t\ u \in [\![(\sigma + x \mapsto u)(\tau_2')]\!]$ *because of (4).*

- *Case* (Sub-Set)*: The subtyping derivation ends with* $\Gamma \vdash \{x : \theta \mid t\} \leq \{x : \theta \mid u\}$ *and the premises are* $\Gamma, (x : \theta) \vdash t \Rightarrow u$ ***(1)*** *and, again by inversion of (1), we get* $\forall \sigma$, *if* $\vdash \sigma : (\Gamma, (x : \theta))$ ***(2)*** *and* $\sigma(t) \rightsquigarrow^{\star} \mathsf{T}$ *then* $\sigma(u) \rightsquigarrow^{\star} \mathsf{T}$ ***(3)***. *Let us take* σ *such that* $\vdash \sigma : \Gamma$ *and a term* t' *such that* $t' \in [\![\sigma(\{x : \theta \mid t\})]\!]$, i.e. $t' \in [\![\{x : \theta \mid \sigma(t)\}]\!]$. *This means that if* $t' \rightsquigarrow^{\star} \boldsymbol{c}$ *then* $\sigma(t)[x \mapsto t'] \rightsquigarrow^{\star} \mathsf{T}$ ***(4)***. *The substitution* $\sigma' \equiv \sigma + x \mapsto t'$ *is such that* $\vdash \sigma' : (\Gamma, (x : \theta))$ *because* $\vdash \sigma : \Gamma$ *and* $\sigma'(x) \equiv t' \in [\![\sigma(\theta)]\!] \equiv [\![\theta]\!] \equiv \{t \mid \vdash_{\mathsf{s}} \lfloor t \rfloor : \theta\}$. *Applying (3) on* σ' *and (4) entails* $\sigma'(u) \rightsquigarrow^{\star} \mathsf{T}$. *Thus, the term* t' *is in* $[\![\sigma(\{x : \theta \mid u\})]\!]$.

On pose les contraintes suivantes sur les définitions des types et des interprétations des constantes pour qu'elles soient compatibles avec le typage sémantique :

We state the following requirements on the definitions of the types and interpretations of constants for them to be compliant with the semantic typing:

Requirement 1 *For all* $\boldsymbol{c}$, $\vdash ty(\boldsymbol{c})$ *holds.*

Requirement 2 *For all* $\boldsymbol{c}, v$, *if* $ty(\boldsymbol{c}) = \Pi(x : \tau_1).\tau_2$ *and if* $v \in [\![\tau_1]\!]$, *then* $\delta(\boldsymbol{c}, v) \in [\![\tau_2[x \mapsto v]]\!]$.

Requirement 3 *For all* $\boldsymbol{c}$, *if* $ty(\boldsymbol{c}) = \{x : \theta \mid t\}$ *then* $t[x \mapsto \boldsymbol{c}] \rightsquigarrow^\star \mathsf{T}$.

Question 14

Montrez que l'interprétation et le type donné à if_τ dans la question 4 valide ces contraintes.

Show that the interpretation and the type assigned to if_τ in question 4 validates these requirements.

Answer: We first have to validate requirement 1.

Notice that the following rule is clearly admissible:

$$\frac{\Gamma \vdash t : \textbf{\textit{bool}} \qquad \Gamma \vdash u : \textbf{\textit{bool}}}{\Gamma \vdash t \Leftrightarrow u : \textbf{\textit{bool}}}$$

Assuming that τ *is closed, the following type is closed:*

$$\Pi(x : \textbf{\textit{bool}}).(\{u : \textbf{\textit{unit}} \mid x \Leftrightarrow \mathsf{T}\} \to \tau) \to (\{u : \textbf{\textit{unit}} \mid x \Leftrightarrow \mathsf{F}\} \to \tau) \to \tau$$

because of the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{(x : \boldsymbol{b}), (u : \boldsymbol{u}) \vdash x : \boldsymbol{b}}{(x : \boldsymbol{b}), (u : \boldsymbol{u}) \vdash \mathsf{T} : \boldsymbol{b}}}{(x : \boldsymbol{b}), (u : \boldsymbol{u}) \vdash x \Leftrightarrow \mathsf{T} : \boldsymbol{b}} \quad \dfrac{}{(x : \boldsymbol{b}) \vdash \tau}\,(\textit{Hypothesis})}{(x : \boldsymbol{b}) \vdash \{u : \boldsymbol{u} \mid x \Leftrightarrow \mathsf{T}\}} \quad \dfrac{}{(x : \boldsymbol{b}) \vdash \tau}\,(\textit{Hypothesis})}{(x : \boldsymbol{b}) \vdash \{u : \boldsymbol{u} \mid x \Leftrightarrow \mathsf{T}\} \to \tau} \qquad \dfrac{\dfrac{\dfrac{\dfrac{(x : \boldsymbol{b}), (u : \boldsymbol{u}) \vdash x : \boldsymbol{b}}{(x : \boldsymbol{b}), (u : \boldsymbol{u}) \vdash \mathsf{F} : \boldsymbol{b}}}{(x : \boldsymbol{b}), (u : \boldsymbol{u}) \vdash x \Leftrightarrow \mathsf{F} : \boldsymbol{b}} \quad \dfrac{}{(x : \boldsymbol{b}) \vdash \tau}\,(\textit{Hypothesis})}{(x : \boldsymbol{b}) \vdash \{u : \boldsymbol{u} \mid x \Leftrightarrow \mathsf{F}\}} \quad \dfrac{}{(x : \boldsymbol{b}) \vdash \tau}\,(\textit{Hypothesis})}{(x : \boldsymbol{b}) \vdash \{u : \boldsymbol{u} \mid x \Leftrightarrow \mathsf{F}\} \to \tau}$$

$$\dfrac{\dfrac{\emptyset \vdash \boldsymbol{b} \qquad (x : \boldsymbol{b}) \vdash (\{u : \boldsymbol{u} \mid x \Leftrightarrow \mathsf{T}\} \to \tau) \to (\{u : \boldsymbol{u} \mid x \Leftrightarrow \mathsf{F}\} \to \tau) \to \tau}{(x : \boldsymbol{b}) \vdash (\{u : \boldsymbol{u} \mid x = \mathsf{T}\} \to \tau) \to (\{u : \boldsymbol{u} \mid x = \mathsf{F}\} \to \tau) \to \tau}}{\emptyset \vdash \Pi(x : \boldsymbol{b}).(\{u : \boldsymbol{u} \mid x = \mathsf{T}\} \to \tau) \to (\{u : \boldsymbol{u} \mid x = \mathsf{F}\} \to \tau) \to \tau}$$

This validates requirement 1.

Given the shape of this type, we then have to validate the requirement 2. Let us take $v \in [\![\textbf{\textit{bool}}]\!]$, *then* v *is* T *or* F. *We only show the case where* $v = \mathsf{T}$*—the other case follows by symmetry. We have to show that* $\delta(\text{if}_\tau, \mathsf{T})$, *which is equal to*

$$\lambda(x : \{u : \textbf{\textit{unit}} \mid \mathsf{T} = \mathsf{T}\} \to \tau).\, \lambda(y : \{u : \textbf{\textit{unit}} \mid \mathsf{T} = \mathsf{F}\} \to \tau).\, x\ ()$$

is an inhabitant of

$$[\![(\{u : \textbf{\textit{unit}} \mid \mathsf{T} = \mathsf{T}\} \to \tau) \to (\{u : \textbf{\textit{unit}} \mid \mathsf{T} = \mathsf{F}\} \to \tau) \to \tau]\!]$$

Asume $t \in [\![\{u : \textbf{\textit{unit}} \mid \mathsf{T} = \mathsf{T}\} \to \tau]\!]$ ***(1)*** *and* $u \in [\![\{u : \textbf{\textit{unit}} \mid \mathsf{F} = \mathsf{T}\} \to \tau]\!]$. *We have* $() \in [\![\{u : \textbf{\textit{unit}} \mid \mathsf{T} = \mathsf{T}\}]\!]$ *because* $\mathsf{T} = \mathsf{T} \rightsquigarrow^\star \mathsf{T}$. *Thus by (1) we have* $t\ () \in [\![\tau]\!]$, *as expected.*

Question 15

Prouvez que votre définition de $[\![\bullet]\!]$ vérifie la seconde propriété du Lemme 4. Traitez uniquement le cas des règles (B-Var), (Cst) et (Sub).

Prove that your definition of $[\![\bullet]\!]$ verifies the second property of Lemma 4. Give only the cases for the rules (B-Var), (Cst) and (Sub).

Answer: We must show that $\Gamma \vdash t : \tau$ ***(1)*** *implies* $\Gamma \vdash t \in \tau$ ***(2)***. *The proof is by induction over the typing derivation of the (1). In each case, we establish (2) by* Semantic-Subtyping. *For that purpose, we assume* $\vdash \sigma : \Gamma$ ***(3)*** *and show* $\sigma(t) \in [\![\sigma(\tau)]\!]$ ***(4)***.

- *Case* (B-Var)*: Then (1) is* $\Gamma \vdash x : \{y : \theta \mid t \,\&\, y =_\theta x\}$ *and follows from the premise* $(x : \{y : \theta \mid t\}) \in \Gamma$*. This with (3) implies* $\sigma(x) \in [\![\sigma(\{y : \theta \mid t\})]\!]$*. This means that if* $\sigma(x) \rightsquigarrow^\star \boldsymbol{c}$ *then* $\sigma(t)[y \mapsto \boldsymbol{c}] \rightsquigarrow^\star \mathsf{T}$*. In that case, we also have that* $\sigma((y =_\theta x))[y \mapsto \boldsymbol{c}] = (\boldsymbol{c} =_\theta \sigma(x)) \rightsquigarrow^\star \mathsf{T}$*. Thus we also have* $\sigma(x) \in [\![\sigma(\{y : \theta \mid t \,\&\, y =_\theta x\})]\!]$, i.e. *the goal (4).*

- *Case* (Cst)*: Then (1) is* $\Gamma \vdash \boldsymbol{c} : ty(\boldsymbol{c})$*. Notice that* $\sigma(\boldsymbol{c}) = \boldsymbol{c}$ *and that* $\sigma(ty(\boldsymbol{c})) = ty(\boldsymbol{c})$ *by requirement 1. We have two subcases:*

  - *Case* $ty(\boldsymbol{c}) \equiv \Pi(x : \tau_1).\tau_2$*: The goal (4) is* $\boldsymbol{c} \in [\![\Pi(x : \tau_1).\tau_2]\!]$*. Thus, we must show both* $\vdash_s \lfloor \boldsymbol{c} \rfloor : \lfloor \tau_1 \rfloor \to \lfloor \tau_2 \rfloor$ *and* $\forall u \in [\![\tau_1]\!], \boldsymbol{c}\ u \in [\![\tau_2[x \mapsto u]]\!]$*. The first part follows from (1) and Question 7. To show the second part, let* u *be such that* $u \in [\![\tau_1]\!]$*. Then,* $\vdash_s \lfloor u \rfloor : \lfloor \tau_1 \rfloor$ *holds. This judgement and the first part entails* $\vdash_s \boldsymbol{c}\ y : \lfloor \tau_2 \rfloor$*. If the evaluation of* u *converges to a value* v*, then we conclude by requirement 2; otherwise, the term* $\boldsymbol{c}\ u$ *diverges so it is also in* $[\![\tau_2[x \mapsto u]]\!]$*.*
  - *Case* $ty(\boldsymbol{c}) \equiv \{x : \theta \mid u\}$*: The goal (4) is* $\boldsymbol{c} \in [\![\{x : \theta \mid u\}]\!]$*. That is, we must show both* $\vdash_s \lfloor \boldsymbol{c} \rfloor : \theta$ *and* $u[x \mapsto \boldsymbol{c}] \rightsquigarrow^\star \mathsf{T}$*. The first part follows from (1) and Question 7; the second part follows from requirement 3.*

- *Case* (Sub)*: Then (1) is* $\Gamma \vdash t : \tau_2$ *with the premise* $\Gamma \vdash t : \tau_1$*. By induction,* $\Gamma \vdash t \in \tau_1$ ***(5)****. By the first property of the lemma,* $\Gamma \vdash \tau_1 \leq \tau_2$ *implies* $\Gamma \vdash \tau_1 \subseteq \tau_2$*. So, (5) implies* $\Gamma \vdash t \in \tau_2$ *by unfolding the definition of* Semantic-Subtyping *and* Semantic-Typing*.*

4 Correction du système de type / Type soundness

Dans cette section, vous pouvez utiliser le Lemme 4 même si vous ne l'avez pas prouvé dans les questions précédentes.

In this section, you can use the Lemma 4 even if you did not prove it in the previous questions.

Question 16

Montrer que la relation de sous-typage est stable par substitution :

Show that the subtyping relation is stable by substitution:

Lemma 5 *Assume that* $\Gamma \vdash t : \tau$ *holds.*
For all Γ' *and* x*, if* $\Gamma, (x : \tau), \Gamma' \vdash \tau_1 \leq \tau_2$ *holds then* $\Gamma, \Gamma'[x \mapsto t] \vdash \tau_1[x \mapsto t] \leq \tau_2[x \mapsto t]$ *holds.*

Donnez uniquement le cas de la règle (Sub-Set).

Give only the case for the rule (Sub-Set).

Answer:

By induction over subtyping derivations.

- *Case* (Sub-Set)*: The conclusion is* $\Gamma, (x : \tau), \Gamma' \vdash \{z : \theta \mid u\} \leq \{z : \theta \mid u'\}$*. The hypothesis is* $\Gamma, (x : \tau), \Gamma', (z : \theta) \vdash u \Rightarrow u'$*. Let us write* Γ'' *for* $\Gamma', (z : \theta)$*. By inversion of this judgement, we have that for all* σ *such that* $\vdash \sigma : (\Gamma, (x : \tau), \Gamma'')$ ***(1)****, if* $\sigma(u) \rightsquigarrow^\star \mathsf{T}$ *then* $\sigma(u') \rightsquigarrow^\star \mathsf{T}$*.*

  By the second property of the Lemma 4, $\Gamma \vdash t : \tau$ *implies* $\Gamma \vdash t \in \tau$ ***(2)****.*

  Let us take σ *such that* $\vdash \sigma : \Gamma, \Gamma''[x \mapsto t]$ ***(3)*** *and define* $\sigma' \equiv \sigma + x \mapsto t$*.*

  By (3), for all y*, if* $y \in dom(\sigma)$*, then* $\sigma(y) \in [\![\sigma((\Gamma, \Gamma''[x \mapsto t])(y))]\!]$ ***(4)****.*

  Let us show that for all y*, if* $y \in dom(\sigma')$ *then* $\sigma'(y) \in [\![\sigma'((\Gamma, (x : \tau), \Gamma'')(y))]\!]$ ***(5)****.*

  Consider $y \in dom(\sigma')$*. Either* $y = x$ *or* $y \in dom(\sigma)$*.*

- *Case 'y = x': By definition of σ', $\sigma'(y) = t$. By (2), $t \in [\![\tau]\!]$, i.e. $\sigma'(t) \in [\![\sigma'(\tau)]\!] = [\![\sigma'((\Gamma, (x : \tau), \Gamma'')(x))]\!]$.*
- *Case 'y ∈ dom(σ)': We have $\sigma'(y) = \sigma(y)$ and thus by (4), $\sigma'(y) \in [\![\sigma((\Gamma, \Gamma''[x \mapsto t])(y))]\!]$ (**6**). We must show that $\sigma'(y) \in [\![\sigma'((\Gamma, ((x : \tau)), \Gamma'')(y))]\!]$ (**7**).*
  - *If $y \in dom(\Gamma)$, then (6) is $\sigma'(y) \in [\![\sigma(\Gamma(y))]\!]$ and (7) is $\sigma'(y) \in [\![\sigma'(\Gamma(y))]\!]$. x cannot occur in $\Gamma(y)$, so $\sigma(\Gamma(y)) = \sigma'(\Gamma(y))$.*
  - *If $y \in dom(\Gamma'')$, then (6) is $\sigma'(y) \in [\![\sigma(\Gamma''[x \mapsto t](y))]\!]$ and (7) is $\sigma'(y) \in [\![\sigma'(((x : \tau), \Gamma'')(y))]\!] = [\![\sigma((\Gamma''[x \mapsto t])(y))]\!]$.*

Question 17

Prouvez le cas (Sub) de la preuve par induction mutuelle du lemme suivant.

Prove the case (Sub) of the mutual induction of the proof of the following lemma.

Lemma 6 *Assume that $\Gamma \vdash t : \tau$ holds.*

- *If $\Gamma, (x : \tau), \Gamma' \vdash u : \tau'$ holds then $\Gamma, \Gamma'[x \mapsto t] \vdash u[x \mapsto t] : \tau'[x \mapsto t]$*
- *If $\Gamma, (x : \tau), \Gamma' \vdash \tau'$ holds then $\Gamma, \Gamma'[x \mapsto t] \vdash \tau'[x \mapsto t]$*

Answer:

- *Case (Sub): The conclusion is $\Gamma, (x : \tau), \Gamma' \vdash u : \tau'$. By inversion, $\Gamma, (x : \tau), \Gamma' \vdash u : \tau''$ such that $\Gamma, (x : \tau), \Gamma' \vdash \tau'' \leq \tau'$. Applying the Lemma 5 to this judgement, we get $\Gamma, \Gamma'[x \mapsto t] \vdash \tau''[x \mapsto t] \leq \tau'[x \mapsto t]$. By induction, $\Gamma, \Gamma'[x \mapsto t] \vdash u[x \mapsto t] : \tau''[x \mapsto t]$ and $\Gamma, \Gamma'[x \mapsto t] \vdash \tau'[x \mapsto t]$ We conclude by (Sub).*

Question 18

Énoncez et prouvez la propriété de préservation du typage.

State and prove the property of subject reduction.

Answer:

Lemma 7 (Subject reduction) *If $\vdash t : \tau$ and $t \rightsquigarrow u$ then $\vdash u : \tau$.*

To prove this lemma, we first have to use the result of the course about subtyping saying that if $\vdash t : \tau$ then there exists a proof of it that never uses two consecutive applications of (Sub).
By induction over reductions.

- *Case (Red-β): We have $\vdash (\lambda(x : \tau).\, t)\ v : \tau'[x \mapsto v]$ (**1**) and $(\lambda(x : \tau).\, t)\ v \rightsquigarrow t[x \mapsto v]$. By inversion of (1), $\vdash v : \tau$ and $(x : \tau) \vdash t : \tau'$. By Lemma 6, we have $\vdash t[x \mapsto v] : \tau'[x \mapsto v]$.*
- *Case (Red-δ): We have $\vdash \boldsymbol{c}\ v : \tau_2[x \mapsto v]$ (**1**) and $\boldsymbol{c}\ v \rightsquigarrow \delta(\boldsymbol{c}, v)$. By inspection of the definition of δ, we conclude that $\vdash \delta(\boldsymbol{c}, v)$: has indeed the type $\tau_2[x \mapsto v]$.*
- *Case (Red-Context): We have $\vdash \mathcal{C}[t] : \tau$ (**1**) and by inversion of the reduction, $t \rightsquigarrow u$. A context is an application so the last typing rule used to derive (1) is either (App) or (Sub).*
  - *Case (App): $\mathcal{C} \equiv v\ []$ or $\mathcal{C} \equiv []\ u$. In both case, there are an hypothesis of the form $\vdash t : \tau'$ and another hypothesis of the form $\vdash t' : \tau''$ (**1**). By induction, $\vdash u : \tau'$. By application of (App) to this judgment and (1), we get $\vdash \mathcal{C}[u] : \tau$.*
  - *Case (Sub): By inversion, $\vdash t : \tau'$ (**1**) for some type τ' such that $\vdash \tau' \leq ty$. We proceed as in the previous case on (1): thanks to our initial remark, we now that is must be an application of (App).*