Programmation fonctionnelle et systèmes de types (MPRI 2-4-2)
Examen partiel 2009–2010

François Pottier

17 novembre 2009

Ce problème étudie la question de la sûreté du typage dans le système F_η. Il est divisé en plusieurs parties. Les parties 1 à 3 sont introductives. Les parties 4 et 5 sont indépendantes l'une de l'autre. La partie 6 fait suite à la partie 5. La partie 7 ne dépend que de la partie 1.

1 Types et coercions

On rappelle la syntaxe des types commune au système F et au système F_η :

$$T ::= X \mid T \to T \mid \forall X.T$$

Les types du système F_η sont reliés par une relation dite d'instance, ou de sous-typage, notée $\leq$. Cette relation est définie par un jeu de règles relativement nombreuses : un jugement de la forme $T_1 \leq T_2$ peut admettre une dérivation complexe. Pour faciliter le raisonnement à propos de ces dérivations, nous introduisons un langage de *coercions*. Une coercion c peut être considérée comme le témoin, ou la preuve, d'une affirmation de la forme $T_1 \leq T_2$. La syntaxe des coercions est la suivante :

$$c ::= \mathit{id} \mid c; c \mid \mathit{intro} \mid @T \mid c \to c \mid \forall X.c \mid \mathit{distrib}$$

On notera qu'une coercion peut mentionner un type (via la forme $@T$), donc peut avoir des variables de types libres. On notera que la forme $\forall X.c$ lie la variable de types X dans la coercion c.

La syntaxe ci-dessus peut sembler mystérieuse, mais la signification de chacune de ces constructions doit s'éclairer lorsque l'on considère les règles de preuve suivantes. Ces règles définissent un jugement $c : T_1 \leq T_2$, que l'on peut lire, par exemple : *la coercion c démontre que T_2 est instance de T_1*. On pourra lire également : *la coercion c convertit une valeur de type T_1 en une valeur de type T_2*.

Réflexivité
$$\mathit{id} : T \leq T$$

Transitivité
$$\frac{c_1 : T_1 \leq T_2 \qquad c_2 : T_2 \leq T_3}{(c_1; c_2) : T_1 \leq T_3}$$

∀-Introduction
$$\frac{X \# T}{\mathit{intro} : T \leq \forall X.T}$$

∀-Elimination
$$@T_2 : \forall X.T_1 \leq [X \mapsto T_2]T_1$$

→-Congruence
$$\frac{c_1 : T_1' \leq T_1 \qquad c_2 : T_2 \leq T_2'}{c_1 \to c_2 : T_1 \to T_2 \leq T_1' \to T_2'}$$

∀-Congruence
$$\frac{c : T_1 \leq T_2}{\forall X.c : \forall X.T_1 \leq \forall X.T_2}$$

Distribution
$$\mathit{distrib} : \forall X.(T_1 \to T_2) \leq (\forall X.T_1) \to (\forall X.T_2)$$

Cette présentation n'est pas exactement celle considérée en cours. Toutefois, les exemples suivants devraient vous convaincre (et on pourrait démontrer) que la relation définie ici est bien identique à celle considérée en cours.

Question 1 *Donner un témoin c de la relation $T_1 \to T_2 \leq (\forall X.X) \to T_2$. On donnera la dérivation du jugement $c : T_1 \to T_2 \leq (\forall X.X) \to T_2$.* ◇

Question 2 *Donner un témoin de la relation $\forall X.\forall Y.(T_1 \to T_2) \leq (\forall X.\forall Y.T_1) \to (\forall X.\forall Y.T_2)$. On ne demande pas de dérivation.* ◇

Question 3 *Donner un témoin de la relation $\forall X.\forall Y.(X \to Y) \leq T_1 \to T_2$. On ne demande pas de dérivation.* ◇

Question 4 *Donner un témoin de la relation $\forall X.(X \to X) \leq \forall Y.\forall Z.((Y \to Z) \to (Y \to Z))$. On ne demande pas de dérivation.* ◇

2 Typage

On donne maintenant la syntaxe des termes du système F_η :

$$t ::= x \mid \lambda x.t \mid t\ t \mid \Lambda X.t \mid c\ t$$

Cette syntaxe est explicitement typée, au sens où Λ-abstractions et coercions sont explicites. Toutefois, pour des raisons de légèreté, les λ-abstractions ne portent pas explicitement le type de leur argument. (Ce choix est quelque peu non standard, mais correspond à ce qui a été présenté en cours.) La construction $c\ t$ doit être lue comme l'application d'une coercion à un terme. On notera que, puisqu'une coercion peut avoir des variables de types libres, il en va de même d'un terme.

Les règles de typage du système F_η sont les suivantes. (Un environnement de typage Γ est une séquence de liaisons de la forme $x : T$.) L'application d'une coercion à un terme, via la règle Sub, permet d'exploiter la relation d'instance. Ainsi, cette présentation du système F_η est dirigée par la syntaxe, ce qui facilitera la preuve de sûreté du typage.

$$\textsc{Var}\ \Gamma \vdash x : \Gamma(x) \qquad \textsc{Abs}\ \frac{\Gamma; x : T_1 \vdash t : T_2}{\Gamma \vdash \lambda x.t : T_1 \to T_2} \qquad \textsc{App}\ \frac{\Gamma \vdash t_1 : T_1 \to T_2 \quad \Gamma \vdash t_2 : T_1}{\Gamma \vdash t_1\ t_2 : T_2}$$

$$\textsc{TAbs}\ \frac{\Gamma \vdash t : T \qquad X \# \Gamma}{\Gamma \vdash \Lambda X.t : \forall X.T} \qquad \textsc{Sub}\ \frac{\Gamma \vdash t : T_1 \qquad c : T_1 \leq T_2}{\Gamma \vdash c\ t : T_2}$$

Question 5 *Pourquoi l'application de types $t ::= \ldots \mid t\ T$, qui dans le système F permet d'éliminer un quantificateur universel, n'apparaît-elle pas ici ? Par quelle construction est-elle remplacée ?* ◇

3 Sémantique

On dote à présent les termes (dont la syntaxe a été donnée ci-dessus) d'une sémantique. Il s'agit d'une sémantique à réduction à petits pas, en appel par valeur.

Pour définir cette sémantique, on spécifie d'abord la syntaxe des valeurs et des contextes d'évaluation :

$$v ::= \lambda x.t \mid \Lambda X.v$$
$$E ::= [] \mid E\ t \mid v\ E \mid \Lambda X.E \mid c\ E$$

Ensuite, la relation de réduction $\longrightarrow$ est définie, de façon inductive, par les règles suivantes :

$$(\lambda x.t)\ v \longrightarrow [x \mapsto v]t \tag{1}$$
$$id\ v \longrightarrow v \tag{2}$$
$$(c_1; c_2)\ v \longrightarrow c_2\ (c_1\ v) \tag{3}$$
$$intro\ v \longrightarrow \Lambda X.v \quad \text{si } X \# v \tag{4}$$
$$(@T)\ (\Lambda X.v) \longrightarrow [X \mapsto T]v \tag{5}$$
$$(c_1 \to c_2)\ (\lambda x.t) \longrightarrow \lambda x.(c_2\ ([x \mapsto c_1\ x]t)) \tag{6}$$
$$(\forall X.c)\ (\Lambda X.v) \longrightarrow \Lambda X.(c\ v) \tag{7}$$
$$distrib\ (\Lambda X.\lambda x.t) \longrightarrow \lambda x.\Lambda X.([x \mapsto @X\ x]t) \tag{8}$$
$$E[t_1] \longrightarrow E[t_2] \quad \text{si } t_1 \longrightarrow t_2 \tag{9}$$

Cette sémantique est « à effacement des types », au sens où, même si les termes contiennent des annotations liées au typage (Λ-abstractions, coercions), et même si les règles de réduction ci-dessus suggèrent que ces annotations jouent un rôle au cours de calcul, ceci n'est pas réellement le cas. On peut en fait effacer ces annotations avant l'exécution sans affecter le résultat du calcul. On pourrait démontrer ce fait en établissant que la fonction d'effacement des annotations est une simulation. Nous ne le ferons pas ici.

4 Relation entre les systèmes F et F_η

On souhaite maintenant préciser la relation entre les systèmes F et F_η.

On rappelle la syntaxe des termes du système F :

$$u ::= x \mid \lambda x.u \mid u\ u \mid \Lambda X.u \mid u\ T$$

Si u est un terme du système F, on note $\lfloor u \rfloor$ le λ-terme pur obtenu à partir de u par effacement de toutes les abstractions et applications de types.

Dans le système F, on ne dispose ni de coercions, ni de la relation d'instance du système F_η : on ne dispose que des constructions classiques d'abstraction et d'application de types. Cependant, une traduction du système F_η dans le système F est possible. La traduction des constructions communes aux systèmes F et F_η est bien sûr triviale : la seule difficulté est de traduire l'application d'une coercion à un terme. Nous considérons à présent cette question.

Question 6 *Pour toute coercion $c : T_1 \leq T_2$, et pour tout terme u du système F tel que $\Gamma \vdash u : T_1$ est dérivable dans le système F, construire un terme du système F, que l'on notera $[\![c\ u]\!]$, de sorte que les deux propriétés suivantes soient satisfaites :*

1. *$\Gamma \vdash [\![c\ u]\!] : T_2$ est dérivable dans le système F ;*
2. *les λ-termes purs $\lfloor u \rfloor$ et $\lfloor [\![c\ u]\!] \rfloor$ sont η-équivalents.*

On se contentera de définir la fonction $[\![\cdot\ \cdot]\!]$. On ne démontrera pas explicitement que ces deux propriétés sont satisfaites. ◇

Il découle de ce résultat que, si $\Gamma \vdash t : T$ est dérivable dans le système F_η, alors il existe un terme u tel que $\lfloor t \rfloor$ et $\lfloor u \rfloor$ sont η-équivalents et tel que $\Gamma \vdash u : T$ est dérivable dans le système F.

En d'autres termes, le système F_η n'est pas beaucoup plus expressif que le système F : il permet simplement d'omettre (ou, plus précisément, de remplacer par des coercions) certaines η-expansions explicites. On peut ainsi espérer gagner un peu en concision (si les coercions ne sont pas données explicitement) et surtout en efficacité (car les coercions sont effacées avant l'exécution, tandis qu'un η-redex pourra donner lieu à du code).

Mitchell [1] a démontré la réciproque de ce résultat, ce qui implique que le système F_η est exactement la « clôture du système F modulo η. »

5 Sûreté du système F_η

On démontre maintenant que la réduction préserve le typage. Comme dans le cours, la preuve comporte un cas par règle de réduction, donc neuf cas en tout. On ne demande de traiter ici **qu'une partie** de ces cas.

Question 7 (Auto-réduction) *Démontrer que, si $\Gamma \vdash t : T$, et si t se réduit en t'* ***via l'une des règles de réduction 4 à 8,*** *alors $\Gamma \vdash t' : T$. Dans chaque cas, on analysera la forme de la dérivation du jugement $\Gamma \vdash t : T$, et on construira une dérivation du jugement $\Gamma \vdash t' : T$. On admettra sans démonstration les lemmes auxiliaires standard (substitution, etc.). On prendra toutefois soin de donner avec précision les énoncés de ces lemmes.* ◇

On démontre ensuite la propriété de progrès. Comme dans le cours, la preuve comporte un cas par construction dans la syntaxe des termes du système F_η, donc cinq cas en tout. On ne demande de traiter ici **qu'une partie** de ces cas.

Question 8 (Progrès) *Démontrer qu'un terme* ***de la forme*** *$\Lambda X.t$* ***ou*** *$c\ t$, bien typé dans l'environnement vide, soit se réduit soit est une valeur. On admettra sans démonstration le lemme de classification des valeurs, dont on donnera avec précision l'énoncé.* ◇

6 Ajout des références au système F_η

En vue d'une éventuelle introduction des références dans le langage, on décide de restreindre le polymorphisme aux *pseudo-valeurs*, données par la grammaire $w ::= x \mid \lambda x.t \mid \Lambda X.w \mid c\ w$. La règle de typage TAbs est donc remplacée par la règle suivante :

$$\frac{\Gamma \vdash w : T \qquad X \# \Gamma}{\Gamma \vdash \Lambda X.w : \forall X.T}\ \textsc{TAbs-Pseudo-Value}$$

Question 9 *La propriété de progrès est-elle préservée ? Pourquoi ?* ◇

Question 10 *La propriété d'auto-réduction n'est pas préservée. Donner tous les points où la preuve effectuée lors de la question 7 devient invalide.* ◇

Question 11 *Quelle restriction supplémentaire pourrait-on apporter au système pour restaurer la propriété d'auto-réduction ? (On ne demande pas de refaire la preuve de la propriété d'auto-réduction une fois cette restriction effectuée.)* ◇

Question 12 *On considère le système F_η, dans lequel on remplace* TAbs *par* TAbs-Pseudo-Value*, et auquel on ajoute les trois opérations primitives pour allouer, lire et modifier une référence, dotées de leurs types standard. (On pourra également ajouter des constantes entières et booléennes, ainsi que les opérations primitives associées.) Démontrer que ce système n'est pas sûr, en exhibant un programme bien typé qui plante.* ◇

7 Une variante de la règle de distribution

On se pose la question de savoir si l'on pourrait simplifier la règle Distribution. On considère donc les deux règles suivantes :

Distribution-Gauche
$$distrib_g : \forall X.(T_1 \to T_2) \leq (\forall X.T_1) \to T_2$$

Distribution-Droite
$$\frac{X \# T_1}{distrib_d : \forall X.(T_1 \to T_2) \leq T_1 \to (\forall X.T_2)}$$

On remarque d'abord que la première de ces deux règles n'a guère d'intérêt, car elle n'apporte aucune expressivité nouvelle :

Question 13 *Démontrer que, dans le système F_η privé de la règle* Distribution*, la relation $\forall X.(T_1 \to T_2) \leq (\forall X.T_1) \to T_2$ est satisfaite. Exhiber un témoin de cette relation.* ◇

On vérifie ensuite que les règles Distribution-Droite et Distribution sont en fait équivalentes :

Question 14 *Démontrer que, dans le système F_η privé de la règle* Distribution *mais doté de la règle* Distribution-Droite*, la relation $\forall X.(T_1 \to T_2) \leq (\forall X.T_1) \to (\forall X.T_2)$ est satisfaite. Exhiber un témoin de cette relation. Réciproquement, démontrer que, dans le système F_η, la relation $\forall X.(T_1 \to T_2) \leq T_1 \to (\forall X.T_2)$ est satisfaite lorsque $X \# T_1$. Exhiber un témoin de cette relation.* ◇

Références

[1] John C. Mitchell. Polymorphic type inference and containment. *Information and Computation*, 76(2–3) :211–249, 1988.