Types récursifs. Relations logiques.

Correction.

Documents autorisés (en particulier le poly).

Les questions sont annotées par un niveau de difficulté, variant de (0) (facile) à (3) (difficile).

En comptant 5 minutes passées par problème facile, 10 minutes par problème de difficulté (1), 15 minutes par problème de difficulté (2), et 20 minutes par problème de difficulté (3), vous aurez besoin de 2h05 pour tout faire, ce qui vous laisse le temps nécessaire pour lire ce très long énoncé . . .

Mea culpa : chacun des deux problèmes contient une erreur. Dans la partie I (types récursifs), il est nécessaire de démontrer les questions 3 et 4 par une récurrence simultanée. Ceux qui l'ont vu ont eu droit à un bonus.

Plus grave, dans la partie II (théorème de Friedman), la question 4 est fausse : la relation indiquée n'est pas en général une relation logique. Je donnerai une démonstration correcte dans le corrigé ci-dessous.

Types récursifs

On se propose d'examiner une variante du système des types simples autorisant d'écrire des *types récursifs*. C'est déjà ce que propose CaML, où un type défini peut être défini en fonction de lui-même, par exemple :

```
type arbre = FEUILLE of int
           | NOEUD of arbre * arbre;;
type paradox = P of paradox -> paradox;;
type nat = O
         | S of nat;;
```

On va, pour représenter cette récursivité, ajouter au langage des types simples une construction $\mu X \cdot f(X)$, qui dénote en gros l'unique type τ que l'on définirait en CaML en écrivant `type` τ `=` $f(\tau)$`;;`. Par exemple, on coderait les types ci-dessus sous la forme de `arbre` $= \mu X \cdot$ `int` $+ (X \times X)$, `paradox` $= \mu X \cdot X \Rightarrow X$, `nat` $= \mu X \cdot$ `unit` $+ X$.

Formellement, les types sont :

$$F ::= b | X | F \Rightarrow G | \mu X \cdot F$$

où les b sont des types de base, les X des variables de type et où la variable de type X est liée dans $\mu X \cdot F$. On supposera que l'α-renommage est appliqué aux types implicitement.

On note $\sim$ la plus petite congruence sur les types telle que $\mu X \cdot F \sim F[X := \mu X \cdot F]$. (On rappelle qu'une congruence est une relation d'équivalence qui passe au contexte.)

Les règles de typage sont celles des types simples plus une règle portant sur les types récursifs. Appelons le système suivant *système* μ :

$$\frac{}{\Gamma, x : F \vdash x : F}\,(Var)$$

$$\frac{\Gamma \vdash u : F_1 \Rightarrow F_2 \quad \Gamma \vdash v : F_1}{\Gamma \vdash uv : F_2}\,(App) \qquad \frac{\Gamma, y : F_1 \vdash u[x := y] : F_2}{\Gamma \vdash \lambda x \cdot u : F_1 \Rightarrow F_2}\,(Abs)$$

$$\frac{\Gamma \vdash u : F \quad F \sim G}{\Gamma \vdash u : G}\,(TEq)$$

On ne considérera que la règle β, pas η, comme règle de réduction du λ-calcul dans cette partie.

1. (1) Montrer que tout λ-terme (non typé) u est de type $\mu X \cdot X \Rightarrow X$ dans n'importe quel contexte Γ envoyant chaque variable libre dans u vers $\mu X \cdot X \Rightarrow X$ dans le système μ.

   Par récurrence structurelle sur le λ-terme u. Si u est une variable, c'est par hypothèse sur Γ. Les applications sont typées comme suit :

$$\dfrac{\dfrac{\begin{matrix}\vdots \\ \Gamma \vdash u : \mu X \cdot X \Rightarrow X\end{matrix}}{\Gamma \vdash u : (\mu X \cdot X \Rightarrow X) \Rightarrow (\mu X \cdot X \Rightarrow X)}\,(TEq) \qquad \begin{matrix}\vdots \\ \Gamma \vdash v : \mu X \cdot X \Rightarrow X\end{matrix}}{\Gamma \vdash uv : \mu X \cdot X \Rightarrow X}\,(App)$$

   Les abstractions sont typées comme suit :

$$\dfrac{\dfrac{\begin{matrix}\vdots \\ \Gamma, x : \mu X \cdot X \Rightarrow X \vdash u : \mu X \cdot X \Rightarrow X\end{matrix}}{\Gamma \vdash \lambda x \cdot u : (\mu X \cdot X \Rightarrow X) \Rightarrow (\mu X \cdot X \Rightarrow X)}\,(Abs)}{\Gamma \vdash \lambda x \cdot u : \mu X \cdot X \Rightarrow X}\,(TEq)$$

2. (0) En déduire qu'il existe des λ-termes non fortement normalisants, et même non faiblement normalisants, dans le système μ. (On exhibera un contre-exemple.)

   Le terme $\delta\delta$, avec $\delta \hat{=} \lambda x \cdot xx$, est une contre-exemple connu en λ-calcul non typé, donc aussi en système μ.

3. (3) On va réparer ce défaut en considérant un sous-système μ^+ du système μ, où dans $\mu X \cdot F$ la variable X ne peut avoir que des occurrences positives dans F. (Informellement, si l'on voit les types comme des formules, les occurrences positives sont celles qui apparaissent sous un nombre pair de négations, chaque implication $F \Rightarrow G$ comptant comme une négation de F.)

   Formellement, on dit que X *apparaît positivement* dans une variable de type Y si et seulement si $X = Y$; dans $F \Rightarrow G$ si X apparaît positivement dans G ou négativement dans F; dans $\mu Y \cdot F$ ($Y \neq X$) si X apparaît positivement dans F. Et que, symétriquement, X *apparaît négativement* dans un type si et seulement si : ce type est $F \Rightarrow G$ et X apparaît négativement dans G ou positivement dans F; ou ce type est $\mu Y \cdot F$ ($Y \neq X$) et X apparaît négativement dans F.

   Le système μ^+ est la restriction du système μ où, dans tout type $\mu X \cdot F$, X n'apparaît pas négativement dans F.

   On rappelle qu'un *candidat de réductibilité* est un ensemble de λ-termes S tel que :

   (CR1) Si $u \in S$, alors $u \in SN$.
   (CR2) Si $u \in S$ et $u \to u'$, alors $u' \in S$.
   (CR3) Si u est neutre et pour tout u' tel que $u \to u'$, $u' \in S$, alors $u \in S$.

   On admettra d'autre part le théorème suivant, qui est une instance du théorème de point fixe de Tarski :
   Théorème (Tarski) Soit CR l'ensemble de tous les candidats de réductibilité, $f : CR \to CR$ une fonction monotone ($A \subseteq B \Rightarrow f(A) \subseteq f(B)$), alors f a un *plus petit point fixe* $lfp(f)$. (On rappelle qu'un point fixe de f est un candidat A tel que $f(A) = A$.)
   De plus, si f et g sont deux fonctions monotones telles que $f(A) \subseteq g(A)$ pour tout $A \in CR$, alors $lfp(f) \subseteq lfp(g)$.

   On définit alors une notion RED_F adaptée. Pour tout *contexte de candidats* C qui à chaque variable de type X associe un candidat de réductibilité, pour tout type F de μ^+, on définit RED_F^C par :

$$\begin{array}{rcl}
RED_b^C & \hat{=} & SN \\
 & & \text{(ensemble des termes fortement normalisants)} \\
RED_X^C & \hat{=} & C(X) \\
RED_{F \Rightarrow G}^C & \hat{=} & \{u \mid \forall v \in RED_F^C \cdot uv \in RED_G^C\} \\
RED_{\mu X \cdot F}^C & \hat{=} & lfp(S \text{ candidat} \mapsto RED_F^{C[X:=S]})
\end{array}$$

Montrer que RED_F^C est bien défini. (La difficulté principale est de montrer que (S candidat $\mapsto RED_F^{C[X:=S]}$) est bien monotone. On montrera donc simultanément que : (1) RED_F^C est bien défini, (2) si X n'apparaît pas négativement dans F alors RED_F^C est monotone en $C(X)$, au sens où pour tout S tel que $C(X) \subseteq S$, $RED_F^C \subseteq RED_F^{C[X:=S]}$, (3) si X n'apparaît pas positivement dans F alors RED_F^C est antitone en $C(X)$, au sens où pour tout S tel que $C(X) \subseteq S$, $RED_F^C \supseteq RED_F^{C[X:=S]}$.)

On montre (1)–(3) simultanément par récurrence structurelle sur F. Nous inclerons dans (1) le fait de montrer que REF_F^C est un candidat.

Si F est un type de base b, c'est évident. Notamment, SN est un candidat, comme il l'a déjà été remarqué dans le cours.

Si F est une variable Y, (1) est clair. Si $X \neq Y$, (2) et (3) sont triviaux. Si $Y = X$, X apparaît positivement mais pas négativement dans Y : (2) est clair, et (3) est trivialement vrai (hypothèse "X n'apparaît pas positivement" fausse).

Si F est de la forme $G \Rightarrow H$, alors (1) est clair : REF_F^C est bien défini, et est un candidat par les arguments usuels déjà utilisés dans la preuve de normalisation forte du système F_2 du cours. Montrons (2) : supposons que X n'apparaisse pas négativement dans $G \Rightarrow H$, autrement dit X n'apparaît pas positivement dans G et pas négativement dans H. Pour tout $u \in RED_F^C$, pour tout candidat S tel que $C(X) \subseteq S$, on souhaite montrer que $u \in RED_F^{C[X:=S]}$. Soit donc v quelconque dans $RED_G^{C[X:=S]}$: par hypothèse de récurrence (3), puisque X n'apparaît pas positivement dans G, v est dans RED_G^C. Comme $u \in RED_F^C$, il s'ensuit que $uv \in RED_H^C$. Par hypothèse de récurrence (2), puisque X n'apparaît pas négativement dans H, $uv \in RED_H^{C[X:=S]}$. Comme v est quelconque, u est donc dans $RED_F^{C[X:=S]}$, et (2) est prouvé. La preuve de (3) est symétrique.

Si F est de la forme $\mu Y \cdot G$, alors par l'hypothèse de récurrence (2) le théorème de Tarski s'applique et (S candidat $\mapsto RED_G^{C[Y:=S]}$) a donc un plus petit point fixe dans le treillis complet des candidats : ceci prouve (1), notamment que RED_F^C est un candidat. Montrons (2) : supposons que X n'apparaisse pas négativement dans $\mu Y \cdot G$, autrement dit X n'apparaît pas négativement dans G. Par α-renommage, on peut supposer $X \neq Y$. Soit d'autre part $S' \supseteq C(X)$.

Alors la fonction $f \hat{=} (S$ candidat $\mapsto RED_G^{C[Y:=S]})$ est inférieure point à point à $g \hat{=} (S$ candidat $\mapsto RED_G^{C[X:=S', Y:=S]})$, donc par la deuxième partie du théorème de Tarski $lfp(f) \subseteq lfp(g)$, autrement dit $RED_F^C \subseteq RED_F^{C[X:=S']}$, ce qui prouve (2). Le cas (3) est symétrique.

4. (1) Justifier que RED_F^C est un candidat de réductibilité pour tout contexte de candidats C et pour tout type F. (Au lieu de refaire la preuve complètement, on pourra pour certains des cas à examiner indiquer où ils ont déjà été traités dans le cours.)

   Par récurrence structurelle sur F. Les cas où F est un type de base, une variable de type ou une implication sont comme dans la preuve correspondante pour le système F_2. Le cas où F est de la forme $\mu X \cdot G$ définit RED_F^C comme un plus petit point fixe dans l'espace CR des candidats, et est donc un candidat par construction. (Fin de la preuve.)

   Les élèves perspicaces auront deviné que j'ai un peu triché dans ce dernier cas, pour leur faciliter la tâche. La tricherie est que le théorème de Tarski s'applique à condition que l'on sache que CR muni de $\subseteq$ est un treillis complet. Or c'est vrai, mais je vous ai épargné la tâche d'en trouver une preuve. Pour cela, il suffit de montrer que c'est un inf-demi-treillis complet, autrement dit que tout famille F d'éléments de CR a une borne inférieure. (En effet, une construction algébrique standard montre qu'une fois qu'on a toutes les bornes inférieures, on peut obtenir les bornes supérieures de familles F d'éléments comme étant les bornes inférieures de l'ensemble des majorants de F.) Ici, il est facile de voir que $\bigcap F$ est la borne inférieure demandée, car toute intersection de candidats est clairement un candidat.

   Au passage, si vous êtes intéressé par la démonstration du théorème de Tarski, il y en a une un peu cryptique, mais très élégante et très courte. Soit f une fonction monotone d'un inf-demi-treillis $(E, \leq)$ vers lui-même : le théorème de Tarski dit que f a un plus petit point fixe.

   Soit inf *la fonction borne inférieure. On considère l'ensemble P des* post-points fixes *de f : $P \hat{=} \{x \mid f(x) \leq x\}$. Posons $a \hat{=} \inf P$. Alors comme $a \leq x$ pour tout x dans P, $f(a) \leq f(x)$ pour tout x dans P, puisque f est* *monotone. Mais si x est dans P, $f(x) \leq x$ par définition de P. Donc $f(a) \leq x$ pour tout x dans P. Donc $f(a) \leq \inf P$, c'est-à-dire $f(a) \leq a$, par définition de a. (Autrement dit, a est un post-point fixe de f.) Mais $f(a) \leq a$ implique $f(f(a)) \leq f(a)$ par monotonie de f, donc $f(a)$ aussi est un post-point fixe de f : donc $f(a) \in P$. Mais comme $a = \inf P$, $a \leq f(a)$. Comme on avait déjà $f(a) \leq a$, on en déduit $f(a) = a$: a est un point fixe de f.*

Ensuite, a est le plus petit point fixe de f, car si b est un autre point fixe, b est en particulier un post-point fixe de f, donc est $\geq a$ puisque $a = \inf P$.

Finalement, ce que j'ai présenté dans l'énoncé comme étant la seconde partie du théorème de Tarski est déduite de : si $f(x) \leq g(x)$ pour tout x, alors $lfp(f) \leq lfp(g)$. C'est facile : $lfp(g) = g(lfp(g))$ (puisque c'est un point fixe) $\geq f(lfp(g))$ (par hypothèse), donc $lfp(g)$ est un post-point fixe de f; comme $lfp(f)$ est le plus petit post-point fixe de f, $lfp(f) \leq lfp(g)$.

5. (3) En déduire que tout terme typé dans le système μ^+ est fortement normalisant. (Encore une fois, on pourra se référer à certaines preuves du cours pour les parties de la preuve qui ne changent pas ici.)

   On montre par récurrence structurelle sur la dérivation de typage donnée de $x_1 : F_1, \ldots, x_n : F_n \vdash u : F$ en système μ^+ que pour tout contexte de candidats C, pour tous $v_1 \in RED^C_{F_1}$, $\ldots$, $v_n \in RED^C_{F_n}$, alors $u[x_1 := v_1, \ldots, x_n := v_n] \in RED^C_F$.

   Les cas où la dernière règle appliquée est (Var), (App) ou (Abs) sont comme dans la preuve pour le système F_2, et même comme dans la preuve pour le système des types simples. La seule difficulté est le cas où la dernière règle appliquée est (TEq). Il suffit pour régler ce cas de montrer que $F \sim G$ implique $RED^C_F = RED^C_G$.

   Ceci se démontre par récurrence sur le nombre de règles (réflexivité, symétrie, transitivité, passage au contexte) appliquées pour déduire $F \sim G$. Les cas de la réflexivité, de la symétrie, et de la transitivité sont évidents, ainsi que pour le passage au contexte. Il ne reste donc qu'à démontrer que : $(\dagger)$ $RED^C_{\mu X \cdot F} = RED^C_{F[X:=\mu X \cdot F]}$.

   Je prétends qu'en général : $(**)$ $RED^C_{G[X:=H]} = RED^{C[X:=RED^C_H]}_G$. Alors $(\dagger)$ en est une conséquence facile : $RED^C_{\mu X \cdot F} = lfp(S \text{ candidat} \mapsto RED^{C[X:=S]}_F) = RED^{C[X:=lfp(S \text{ candidat} \mapsto RED^{C[X:=S]}_F)]}_F$ (... car c'est un point fixe) $= RED^{C[X:=RED^C_{\mu X \cdot F}]}_F = RED^C_{F[X:=\mu X \cdot F]}$ (par $(**)$).

   Montrons d'abord, comme dans le point $(*)$ de la preuve de normalisation forte du système F_2 du cours, que : $(*)$ si Y n'est pas libre dans H, alors $RED^C_H = RED^{C[Y:=S]}_H$ pour tout candidat S. C'est évident lorsque H est un type de base, et est comme dans le cours pour les types variables ou implications. Lorsque H est de la forme $\mu Z \cdot K$, $RED^{C[Y:=S]}_H = lfp(S' \text{ candidat} \mapsto RED^{C[Y:=S][Z:=S']}_K) = lfp(S' \text{ candidat} \mapsto RED^{C[Z:=S'][Y:=S]}_K) = lfp(S' \text{ candidat} \mapsto RED^{C[Z:=S']}_K)$ (car Y n'est pas libre non plus dans K, et l'on peut donc appliquer l'hypothèse de récurrence) $= RED^C_H$.

   Il ne reste donc finalement qu'à démontrer $(**)$, ce que l'on fait par récurrence structurelle sur G. Le cas où G est un type de base est évident (les deux côtés sont SN). Les cas où G est une variable de type ou une implication sont comme dans le point $(**)$ de la preuve de normalisation forte du système F_2 du cours. Le cas où G est de la forme $\mu Y \cdot G'$ se traite comme suit : $RED^C_{G[X:=H]} = lfp(S \text{ candidat} \mapsto RED^{C[Y:=S]}_{G'[X:=H]}) = lfp(S \text{ candidat} \mapsto RED^{C[Y:=S][X:=RED^{C[Y:=S]}_H]}_{G'})$ (par hypothèse de récurrence) $= lfp(S \text{ candidat} \mapsto RED^{C[Y:=S, X:=RED^C_H]}_{G'})$ (par $(*)$) $= RED^{C[X:=RED^C_H]}_G$.

Théorème de Friedman

On considère dans cette partie le λ-calcul typé dans le système des types simples. Rappelons que les termes sont :

$$u, v, \ldots ::= x \mid \lambda x \cdot u \mid uv$$

et que les règles de typage sont :

$$\frac{}{\Gamma, x : F \vdash x : F}\,(Var)$$

$$\frac{\Gamma \vdash u : F_1 \Rightarrow F_2 \quad \Gamma \vdash v : F_1}{\Gamma \vdash uv : F_2}\,(App) \qquad \frac{\Gamma, y : F_1 \vdash u[x := y] : F_2}{\Gamma \vdash \lambda x \cdot u : F_1 \Rightarrow F_2}\,(Abs)$$

Une *structure applicative standard* est donnée par une famille d'ensembles E_b, pour tout type de base b. On définit alors E_F pour tout type F par récurrence structurelle sur F comme suit: quand $F = b$, E_b est donné, et quand $F = F_1 \Rightarrow F_2$, E_F est l'ensemble de toutes les applications (=fonctions totales) de E_{F_1} vers E_{F_2}.

Un *environnement* ρ est une fonction qui à chaque variable x associe un élément de $\bigcup_F E_F$.

Un environnement *respecte* un contexte de type $\Gamma \hat{=} x_1 : F_1, \ldots, x_n : F_n$ si et seulement si $\rho(x_i) \in E_{F_i}$ pour tout i, $1 \leq i \leq n$.

On définit une interprétation, dite *standard*, des λ-termes simplement typés en interprétant l'application syntaxique du λ-calcul comme étant la véritable application des fonctions à leurs arguments. Formellement, pour tout contexte Γ, pour tout environnement ρ respectant Γ, pour toute dérivation de typage π de $\Gamma \vdash u : F$, on définit $[\![\pi]\!]\rho$ par récurrence structurelle sur π :

$$\begin{aligned}
\left[\!\!\left[\frac{}{\Gamma, x : F \vdash x : F}\,(Var)\right]\!\!\right]\rho \quad &\hat{=} \quad \rho(x) \\
\left[\!\!\left[\frac{\begin{array}{c}\vdots\,\pi_1 \\ \Gamma \vdash u : F_1 \Rightarrow F_2\end{array} \quad \begin{array}{c}\vdots\,\pi_2 \\ \Gamma \vdash v : F_1\end{array}}{\Gamma \vdash uv : F_2}\,(App)\right]\!\!\right]\rho \quad &\hat{=} \quad f(v), \text{ où } f \hat{=} \left[\!\!\left[\begin{array}{c}\vdots\,\pi_1 \\ \Gamma \vdash u : F_1 \Rightarrow F_2\end{array}\right]\!\!\right]\rho, \quad v \hat{=} \left[\!\!\left[\begin{array}{c}\vdots\,\pi_2 \\ \Gamma \vdash v : F_1\end{array}\right]\!\!\right]\rho \\
\left[\!\!\left[\frac{\begin{array}{c}\vdots\,\pi_1 \\ \Gamma, x : F_1 \vdash u : F_2\end{array}}{\Gamma \vdash \lambda x \cdot u : F_1 \Rightarrow F_2}\,(Abs)\right]\!\!\right]\rho \quad &\hat{=} \quad \text{la fonction qui à } v \in E_{F_1} \text{ associe } \left[\!\!\left[\begin{array}{c}\vdots\,\pi_1 \\ \Gamma, x : F_1 \vdash u : F_2\end{array}\right]\!\!\right](\rho[x := v])
\end{aligned}$$

Par abus de langage, on résumera parfois une dérivation π de $\Gamma \vdash u : F$ au seul terme de preuve u, ce qui permettra d'écrire $[\![x]\!]\rho = \rho(x)$, $[\![uv]\!]\rho = [\![u]\!]\rho([\![v]\!]\rho)$, $[\![\lambda x \cdot u]\!]\rho = (v \mapsto [\![u]\!](\rho[x := v]))$.

Il est évident (on ne demande pas de le démontrer, mais vous devriez vous en convaincre) que cette interprétation est *correcte*, au sens où :

(i) si π dérive $\Gamma \vdash u : F$, et si ρ respecte Γ, alors $[\![\pi]\!]\rho$ est un élément de E_F;

(ii) si π et π' sont deux preuves $\beta\eta$-convertibles, alors $[\![\pi]\!]\rho = [\![\pi']\!]\rho$ pour tout ρ.

On va démontrer ici une forme de réciproque au point (ii). C'est un théorème de complétude dû à Harvey Friedman.

Pour ceci, on considère des familles de relations binaires R_F indicées par des types F. Informellement, pour chaque F, R_F est une relation entre termes d'une part et éléments de E_F d'autre part. On notera $u\ R_F\ a$ pour dire que u et a sont en relation par R_F.

On dit que $R \hat{=} (R_F)_{F \text{ type}}$ est une *relation logique* si et seulement si d'une part elle est compatible avec la convertibilité, autrement dit $u =_{\beta\eta} v$ et $u\ R_F\ a$ implique $v\ R_F\ a$; et si d'autre part:

Pour tous types F_1 et F_2, pour tout terme u, pour tout $f \in E_{F_1 \Rightarrow F_2}$, $u\ R_{F_1 \Rightarrow F_2}\ f$
si et seulement si
pour tout terme v, pour tout a in E_{F_1} tels que $v\ R_{F_1}\ a$, on a $uv\ R_{F_2}\ f(a)$.

Plus succinctement, R est une relation logique si et seulement si

$$u\ R_F\ a \wedge u =_{\beta\eta} v \quad \Rightarrow \quad v\ R_F\ a \tag{1}$$
$$u\ R_{F_1 \Rightarrow F_2}\ f \quad \Leftrightarrow \quad (\forall v, a \cdot v\ R_{F_1}\ a \Rightarrow uv\ R_{F_2}\ f(a)) \tag{2}$$

(On remarquera la similarité avec la notion de réductibilité.)

1. (2) Montrer que si $x_1 : F_1, \ldots, x_n : F_n \vdash u : F$ est dérivable, et si $v_i \; R_{F_i} \; a_i$ pour tout i, $1 \leq i \leq n$, alors $u[x_1 := v_1, \ldots, x_n := v_n] \; R_F \; [\![u]\!][x_1 := a_1, \ldots, x_n := a_n]$. (On rappelle que $[x_1 := v_1, \ldots, x_n := v_n]$ dénote la substitution envoyant chaque x_i vers v_i, et que $[x_1 := a_1, \ldots, x_n := a_n]$ est l'environnement envoyant chaque x_i vers a_i.)

   Par récurrence structurelle sur u (ou plutôt, si l'on veut être formel, sur la dérivation donnée de $x_1 : F_1, \ldots, x_n : F_n \vdash u : F$).

   Si u est une variable, alors c'est l'un des x_i, et ce qu'il faut démontrer est $v_i \; R_F \; a_i$. Or $F = F_i$ par typage, et $v_i \; R_{F_i} \; a_i$ est justement une des hypothèses que l'on a faites.

   Si u est de la forme $u_1 u_2$, avec $x_1 : F_1, \ldots, x_n : F_n \vdash u_1 : G \Rightarrow F$ et $x_1 : F_1, \ldots, x_n : F_n \vdash u_2 : G$, alors par hypothèse de récurrence :

$$\begin{array}{llcl} (a) & u_1[x_1 := v_1, \ldots, x_n := v_n] & R_{G \Rightarrow F} & [\![u_1]\!][x_1 := a_1, \ldots, x_n := a_n] \\ (b) & u_2[x_1 := v_1, \ldots, x_n := v_n] & R_G & [\![u_2]\!][x_1 := a_1, \ldots, x_n := a_n] \end{array}$$

   Par l'équivalence (2) vue comme une implication de gauche à droite, (a) et (b) entraînent :

$$\begin{array}{ll} (c) & (u_1[x_1 := v_1, \ldots, x_n := v_n])(u_2[x_1 := v_1, \ldots, x_n := v_n]) \quad R_F \\ & [\![u_1]\!][x_1 := a_1, \ldots, x_n := a_n]([\![u_2]\!][x_1 := a_1, \ldots, x_n := a_n]) \end{array}$$

   mais ceci n'est rien d'autre que :

$$(d) \quad u[x_1 := v_1, \ldots, x_n := v_n] \quad R_F \quad [\![u]\!][x_1 := a_1, \ldots, x_n := a_n]$$

   ce qui était à démontrer.

   Si u est de la forme $\lambda x \cdot u_1$, avec $x_1 : F_1, \ldots, x_n : F_n, x : G \Rightarrow u_1 : H$ et $F = G \Rightarrow H$, alors par hypothèse de récurrence :

$$(e) \quad u_1[x_1 := v_1, \ldots, x_n := v_n, x := v] \quad R_H \quad [\![u_1]\!][x_1 := a_1, \ldots, x_n := a_n, x := a]$$

   pour tous v et a tels que $v \; R_G \; a$. Par (1), ceci implique :

$$(f) \quad uv[x_1 := v_1, \ldots, x_n := v_n] \quad R_H \quad [\![u_1]\!][x_1 := a_1, \ldots, x_n := a_n, x := a]$$

   Par la propriété de cohérence (ii), $[\![u_1]\!][x_1 := a_1, \ldots, x_n := a_n, x := a]$ est égal à $[\![u]\!][x_1 := a_1, \ldots, x_n := a_n](a)$. Donc (f) entraîne :

$$(g) \quad uv[x_1 := v_1, \ldots, x_n := v_n] \quad R_H \quad [\![u]\!][x_1 := a_1, \ldots, x_n := a_n](a)$$

   et ce pour tous v et a tels que $v \; R_G \; a$. Par l'équivalence (2) vue comme une implication de droite à gauche, on a donc :

$$(h) \quad u[x_1 := v_1, \ldots, x_n := v_n] \quad R_{G \Rightarrow H} \quad [\![u]\!][x_1 := a_1, \ldots, x_n := a_n]$$

   ce qui est le résultat souhaité.

2. (0) En déduire que si pour tout F, R_F est une relation *injective modulo $\beta\eta$* (autrement dit $u \; R_F \; a$ et $v \; R_F \; a$ impliquent $u =_{\beta\eta} v$), alors pour tous termes clos u et v de même type F, tels que $[\![u]\!][] = [\![v]\!][]$, on a en fait $u =_{\beta\eta} v$.

   Par la question précédente avec $n = 0$ (puisque u et v sont clos), $u \; R_F \; [\![u]\!][]$ et $v \; R_F \; [\![v]\!][]$. Si $[\![u]\!][] = [\![v]\!][]$, l'injectivité modulo $\beta\eta$ implique donc $u =_{\beta\eta} v$.

3. (2) On note $[u]$ la classe d'équivalence d'un terme quelconque u modulo $\beta\eta$. Pour tout type F, posons Λ_F l'ensemble des termes clos de type F modulo $\beta\eta$; formellement, $\Lambda_F \hat{=} \{[u] \mid \vdash u : F \text{ dérivable}\}$.

   On construit une interprétation standard $(E_F)_{F \text{ type}}$ en posant $E_b \hat{=} \Lambda_b$ pour tout type de base b.

On définit maintenant une famille de fonctions $r_F : E_F \to \Lambda_F$ et $i_F : \Lambda_F \to E_F$ par récurrence sur F comme suit :

$$\begin{array}{rclrcl} r_b(a \in E_b) & \hat{=} & a \in \Lambda_b & i_b([u] \in \Lambda_b) & \hat{=} & [u] \in E_b \\ r_{F \Rightarrow G}(f \in E_{F \Rightarrow G}) & \hat{=} & \lambda x \cdot r_G(f(i_F([x]))) & i_{F \Rightarrow G}([u] \in \Lambda_{F \Rightarrow G}) & \hat{=} & (a \mapsto i_G([u]\ r_F(a))) \end{array}$$

où la notation $\lambda x \cdot [u]$ dénote $[\lambda x \cdot u]$ dans la définition de $r_{F \Rightarrow G}$ (x est une variable quelconque fixée), et où $[u]\ r_F(a)$ (définition de $i_{F \Rightarrow G}$) dénote $[uv]$, en posant $[v] \hat{=} r_F(a)$.

Montrer que (r_F, i_F) définit une *rétraction* de E_F sur Λ_F au sens où $r_F \circ i_F$ est la fonction identité id_{Λ_F}.

Formellement, il est nécessaire d'étendre le résultat à tout Λ_F^Γ, l'espace des classes d'équivalences de termes u tels que $\Gamma \vdash u : F$ est dérivable. Sinon $i_F([x])$ dans la définition de $r_{F \Rightarrow G}$ notamment n'a aucun sens. Au lieu de corriger cette erreur, je donne l'idée de la preuve, et vous l'adapterez pour qu'elle soit pleinement correcte.

La démonstration est par récurrence structurelle sur F. Si F est un type de base, c'est évident. Si F est de la forme $G \Rightarrow H$, alors pour tout $[u] \in \Lambda_F$, $(r_F \circ i_F)([u]) = \lambda x \cdot r_H(i_F([u])(i_G([x]))) = \lambda x \cdot r_H(i_H([u]\ r_G(i_G([x])))) = \lambda x \cdot [u]\ r_G(i_G([x]))$ (parce que (r_G, i_G) est une rétraction par hypothèse de récurrence) $= \lambda x \cdot [u]\ [x]$ (parce que (r_H, i_H) est une rétraction par hypothèse de récurrence) $= \lambda x \cdot [ux]$ (par convention) $= [\lambda x \cdot ux]$ (par convention) $= [u]$ (par η, à condition de choisir x non libre dans u).

4. (2) Montrer que la famille de relations R_F définies par $u\ R_F\ a$ si et seulement si $a = i_F([u])$ est une relation logique.

Malheureusement, ceci est faux. Essayons de faire la démonstration, et voyons où cela bloque.

D'abord, l'équation (1) est claire, car $u\ R_F\ a$ ne dépend que de $[u]$.

Montrons (2). Si $u\ R_{F_1 \Rightarrow F_2}\ f$, c'est que $f = i_{F_1 \Rightarrow F_2}([u])$, c'est-à-dire $f = (a \mapsto i_{F_2}([u]\ r_{F_1}(a)))$, alors pour tous v et a tels que $v\ R_{F_1}\ a$ (autrement dit, $a = i_{F_1}([v])$), on a $f(a) = i_{F_2}([u]\ r_{F_1}(a)) = i_{F_2}([u]\ r_{F_1}(i_{F_1}([v]))) = i_{F_2}([u]\ [v])$ (car (r_{F_1}, i_{F_1}) est une rétraction) $= i_{F_2}([uv])$ (par convention). Mais le fait que $f(a) = i_{F_2}([uv])$ signifie précisément que $uv\ R_{F_2} f(a)$.

Réciproquement, si pour tous v et a tels que $v\ R_{F_1}\ a$ (autrement dit, $a = i_{F_1}([v])$), on a $uv\ R_{F_2} f(a)$, alors $f(a) = i_{F_2}([uv]) = i_{F_2}([u]\ [v]) = i_{F_2}([u]\ r_{F_1}(a))$ (car $r_{F_1}(a) = r_{F_1}(i_{F_1}([v])) = [v]$ puisque (r_{F_1}, i_{F_1}) est une rétraction). On est tenté ici de conclure que $f = (a \mapsto i_{F_2}([u]\ r_{F_1}(a))) = i_{F_1 \Rightarrow F_2}([u])$, c'est-à-dire $u\ R_{F_1 \Rightarrow F_2} f$. Mais ceci demanderait que a soit quelconque, ce qui est faux : a est quelconque dans l'image de i_{F_1}.

On se sortirait de ce mauvais pas en montrant que i_{F_1} est surjective. Mais comme (r_{F_1}, i_{F_1}) est une rétraction, i_{F_1} est déjà injective. Qu'elle soit surjective impliquerait que Λ_{F_1} et E_{F_1} soient de même cardinalité, ce qui est impossible : Λ_{F_1} est toujours dénombrable, alors que E_{F_1} ne l'est plus dès que F_1 est un type flèche.

J'aurais aimé que certains s'aperçoivent du problème et cherchent une solution, mais personne n'a semblé s'en rendre compte. Voici néanmoins une démonstration réparée du théorème de Friedman.

Noter que toute relation logique R est déterminée de façon unique par les R_b, b types de base. R_F est en effet définie de façon unique par récurrence structurelle sur F. Choisissons R telle que $u\ R_b\ a$ si et seulement si $[u] = a$ sur $E_b = \Lambda_b$. Montrons maintenant par récurrence sur F les deux implications:

$$i_F([u]) = f \quad \Rightarrow \quad u\ R_F\ f \tag{3}$$

$$u\ R_F\ f \quad \Rightarrow \quad [u] = r_F(f) \tag{4}$$

Ceci est évident quand F est un type de base. Soit donc F de la forme $G \Rightarrow H$. Alors:

- *(3) : supposons que $i_{G \Rightarrow H}([u]) = f$; par définition $f = (a \mapsto i_H([u]\ r_G(a)))$. Pour montrer que $u\ R_{G \Rightarrow H}\ f$, choisissons v de type G quelconque, $a \in E_G$ quelconque tels que : (i) $v\ R_G a$, il suffit alors de montrer que $uv\ R_H f(a)$. Mais $f(a) = i_H([u]\ r_G(a))$ et par hypothèse de récurrence (4) et (i), $r_G(a) = [v]$, donc $f(a) = i_H([uv])$. Par hypothèse de récurrence (3), $uv\ R_H f(a)$.*
- *(4) : supposons que $u\ R_{G \Rightarrow H} f$, c'est-à-dire que : (ii) pour tout v de type G, pour tout $a \in E_G$ tel que $v\ R_G\ a$, $uv\ R_H f(a)$. Alors $r_F(f) = \lambda x \cdot r_H(f(i_G([x])))$. Mais par hypothèse de récurrence (3) $x\ R_G\ i_G([x])$, donc par (ii) en posant $v \hat{=} x$, $a \hat{=} i_G([x])$, $ux\ R_H\ f(i_G([x]))$. Par hypothèse de récurrence (4) il s'ensuit que $r_H(f(i_G([x]))) = [ux]$, donc $r_F(f) = \lambda x \cdot [ux] = [\lambda x \cdot ux] = [u]$ par η.*

On en conclut le théorème de Friedman : si u et v sont clos de même type F et $[\![u]\!][] = [\![v]\!][]$, alors par la question 1 on a $u\ R_F\ [\![u]\!][]$ et $v\ R_F\ [\![v]\!][]$, donc par (4) $[u] = r_F([\![u]\!][]) = r_F([\![v]\!][]) = [v]$. Lorsque u ou v n'est pas clos, on les clôt d'abord comme à la question suivante.

5. (1) Déduire des questions précédentes qu'il existe une structure applicative standard dans laquelle pour tous termes u et v tels que $\Gamma \vdash u : F$ et $\Gamma \vdash v : F$ soient dérivables, $u =_{\beta\eta} v$ si et seulement si pour tout environnement ρ qui respecte Γ, $[\![u]\!]\rho = [\![v]\!]\rho$.

   Faisons comme si la question précédente était correcte.

   On choisit comme structure applicative standard celle de la question 3. La direction "seulement si" est le fait (ii) énoncé au début de cette partie.

   Pour la direction "si", commençons par traiter le cas où u et v sont clos, et Γ est vide. On a $[\![u]\!][] = [\![v]\!][]$. Mais la relation logique $R \hat{=} (R_F)_{F \texttt{type}}$ de la question 4 est injective modulo $\beta\eta$, puisque $u\ R_F\ a$ et $v\ R_F\ a$ signifient $a = i_F([u]) = i_F([v])$ et impliquent donc $[u] = r_F(i_F([u])) = r_F(i_F([v])) = [v]$. Par la question 2, on a donc $u =_{\beta\eta} v$.

   Dans le cas général, supposons que pour tout environnement ρ qui respecte Γ, $[\![u]\!]\rho = [\![v]\!]\rho$. Posons $\Gamma \hat{=} x_1 : F_1, \ldots, x_n : F_n$, et considérons les termes clos $\lambda x_1 \cdot \ldots \cdot \lambda x_n \cdot u$ et $\lambda x_1 \cdot \ldots \cdot \lambda x_n \cdot v$. Alors $[\![\lambda x_1 \cdot \ldots \cdot \lambda x_n \cdot u]\!][] = [\![\lambda x_1 \cdot \ldots \cdot \lambda x_n \cdot v]\!][]$, donc par l'argument ci-dessus $\lambda x_1 \cdot \ldots \cdot \lambda x_n \cdot u =_{\beta\eta} \lambda x_1 \cdot \ldots \cdot \lambda x_n \cdot v$. Appliquons chaque côté de l'égalité à $x_1, \ldots, x_n$ dans cet ordre : par β-réduction, on obtient $u =_{\beta\eta} v$.