Université Paris Diderot - Master 1 Linguistique Informatique - Sémantique computationnelle - 2014/2015

TD 6 - Lambda calcul non typé

1. **Convention de notation** Donnez les formes entièrement parenthésées des termes suivants. Si ce dont des rédex, les réduire autant que possible.

(1) a. $(\lambda x.\lambda y.fxy)xy$
b. $(\lambda x.\lambda y.xyy)\lambda y.\lambda a.y$

2. **Syntaxe des lambda-termes** Donnez l'arbre de décomposition syntaxique de :

(2) a. $\lambda f.\lambda g.\lambda x.(f)(g)x$
b. $\lambda f.\lambda g.\lambda x.((f)g)x$
c. aab, à comparer avec $(a)(a)b$
d. $(M)ab$, à comparer avec Mab
e. $(\lambda f.(f)g)x$

On rappelle que les étiquettes des noeuds d'un tel arbre peuvent être : LA (lambda abstraction), AF (application fonctionnelle) ou VAR (variable).

Quelle est la forme de l'arbre de décomposition d'une expression béta-réductible ?

3. **β-réduction :** Réduire autant que possible les termes suivants

(3) a. $(\lambda x.xx)\lambda x.x$
b. $((\lambda x.\lambda y.yx)f)\lambda x.x$
c. $(\lambda n.\lambda f.\lambda x.(f)((n)f)x)\lambda fx.fx$

4. **β-réduction :** Réduire autant que possible les expressions suivantes :

(4) a. $(\lambda x.\ (P)x)m$
b. $((\lambda x.\ \lambda Y.\ (Y)x)j)P$
c. $(\lambda x.\forall z.((\lambda y.((K)x)y)z \rightarrow ((R)z)x))j$

En fixant une interprétation des prédicats K, R et P, et des constantes m et j, proposez pour chacune de ces formules une phrase en français ayant les mêmes conditions de vérité.

5. **β-réduction :** Réduire autant que possible l'expression suivante en faisant attention à la portée des lambda abstracteurs [1].

(5) $((\lambda x.\lambda y.x)((\lambda x.\lambda y.y)\lambda x.\lambda y.x)\lambda x.\lambda y.y)\lambda x.\lambda y.y$

6. **β-réduction :** Identifiez tous les rédex de la forme suivante, et réduisez-la autant que possible. Si de nouveaux rédex apparaissent identifiez-les.

(6) $((\lambda S.\lambda V.(S)(V)\lambda Q.(Q)m)\lambda P.(P)j)\lambda O.\lambda y.(O)\lambda z.((kiss)y)z$

1. Indication : vous devez obtenir finalement une expression de la forme $\lambda a.\lambda b.b$

7. **Combinateurs booléens**

On représente les deux valeurs booléennes « vrai » et « faux » par les fonctions à deux arguments suivantes : $\mathsf{T} =_{\text{def}} \lambda x.\lambda y.x$
$\mathsf{F} =_{\text{def}} \lambda x.\lambda y.y$

Ces définitions, conventionnelles, s'expliquent par la forme très simple que reçoit alors la définition par cas :

$\texttt{if}\ P\ \texttt{then}\ Q\ \texttt{else}\ R =_{\text{def}} ((P)Q)R.$

En effet, si P se réduit en T (ie $P \equiv \mathsf{T}$), alors $\texttt{if}\ P\ \texttt{then}\ Q\ \texttt{else}\ R \equiv \mathsf{T}QR \equiv Q$. De même, si $P \equiv \mathsf{F}$, alors on obtient R.

(7) a. Utiliser cette définition par cas pour définir les opérateurs $\neg$, $\wedge$, $\vee$. Vous pouvez éventuellement vous reporter à la section **Aide**.

b. En utilisant les définitions de T et F calculer la valeur de vérité de :

1. $\neg P$ quand P est faux puis quand P est vrai.
2. $P \wedge Q$ quand P et Q sont vrais.
3. $P \vee Q$ quand P et Q sont faux puis quand P est vrai et Q est faux.
4. A quelle formule mettant en jeu un opérateur binaire liant deux propositions P et Q et quelles valeurs pour P et Q correspond l'expression de la question 5 ?

Aide

- **Question 7** : vous trouverez ci-dessous une aide au raisonnement, essayez de trouver les formules grâce à ce raisonnement. Ces formules sont une combinaison d'une ou plusieurs lambda abstraction (des λx) et des valeurs T et F. Si vous ne parvenez pas à trouver les formules à l'aide de ces raisonnements, le résultat est donné en bas de page, essayez de comprendre comment on y est parvenu à partir du raisonnement.

  On peut définir (relativement) aisément les combinateurs booléens en passant par la définition par cas :

  - Pour NOT : $\neg P$ = `if` P `then` F `else` T
    Le lambda terme recherché est une combinaison des valeurs booléennes T et F telle que si $P = T$, on récupère F, sinon on récupère T[2].
  - Pour AND : $P \wedge Q$ = `if` P `then` (`if` Q `then` T `else` F) `else` F[3]
  - Pour OR : $P \vee Q$ = `if` P `then` T `else` (`if` Q `then` T `else` F)[4]

2. $(NOT)P \equiv ((P)F)T)$, alors si $P \equiv \mathsf{T}$, alors $\mathsf{NOT}P \equiv ((P)\mathsf{F})\mathsf{T} \equiv ((\mathsf{T})\mathsf{F})\mathsf{T} \equiv \mathsf{F}$. D'où par λ-abstraction : $\mathsf{NOT} =_{\text{def}} \lambda x.((x)\mathsf{F})\mathsf{T}$.
3. $((AND)P)Q \equiv ((P)((Q)T)F)F$ d'où : $\mathsf{AND} =_{\text{def}} \lambda x.\lambda y((x)((y)T)F)F$
4. $((OR)P)Q \equiv ((P)T)((Q)T)F$, et je vous laisse trouver la formule finale.