# Examen du cours “Introduction à la compilation”
# Seconde session
# Durée: 3 heures

Tout document autorisé.

Le soin apporté à la rédaction et à la présentation ainsi que la rigueur des réponses seront pris en compte dans la notation. Ce sujet se décompose en 2 parties indépendantes. La première est une application directe du cours d'analyse syntaxique. La suivante porte sur l'étude de la traduction du $\lambda$-calcul dans le langage des combinateurs S, K et I.

## 1 Analyse syntaxique (8 points)

Soient un ensemble de non-terminaux $\{S, E, B\}$ et un ensemble de terminaux $\{n, +, *, \$\}$. On définit la grammaire $G$ suivante :

$$\begin{array}{lcl} S & \rightarrow & E\,\$ \\ E & \rightarrow & E\,B\,E \mid n \\ B & \rightarrow & + \mid * \end{array}$$

**Exercice 1**

1. *Pourquoi cette grammaire est-elle ambiguë ? Proposez une façon de résoudre raisonnablement cette ambiguïté à l'aide d'une nouvelle grammaire $G'$ équivalente et non ambiguë.*
2. *Calculez l'automate LR(1) associé à la grammaire $G$.*
3. *Cet automate contient-il des conflits ? Si oui, expliquez-les.*
4. *Calculez l'automate LR(1) associé à votre grammaire $G'$.*
5. *Ce nouvel automate contient-il des conflits ? Si oui, expliquez-les.*
6. *Donnez les étapes de la reconnaissance du mot « $n + n * n\$$ » par l'automate LR(1) de la grammaire $G'$.*

□

# 2 $\lambda$-calcul et combinateurs S, K et I (12 points)

## 2.1 $\lambda$-calcul en stratégie d'appel par valeur

On se donne la syntaxe suivante pour le $\lambda$-calcul sans constante :

| | | | |
|---|---|---|---|
| $e$ | $::=$ | | ***Expression*** |
| | $\mid$ | $x, f, \ldots$ | *Variable* |
| | $\mid$ | $e\ e$ | *Application* |
| | $\mid$ | $\texttt{fun}\ x \Rightarrow e$ | *Fonction* |

L'utilisation du sucre syntaxique "$\texttt{let}\ x = e_1\ \texttt{in}\ e_2$" pour "$(\texttt{fun}\ x \Rightarrow e_2)\ e_1$" est autorisée. On écrira aussi "$\texttt{let}\ f\ x_1\ \ldots\ x_n = e_1\ \texttt{in}\ e_2$" pour l'expression "$\texttt{let}\ f = \texttt{fun}\ x_1 \Rightarrow \ldots \texttt{fun}\ x_n \Rightarrow e_1\ \texttt{in}\ e_2$". On utilisera la méta-variable $v$ pour dénoter les valeurs de ce langage.

On rappelle les règles de sémantique opérationnelles de la stratégie d'évaluation dite "par valeur" :

$$\frac{}{(\texttt{fun}\ x \Rightarrow e)\ v \to e\{x \mapsto v\}} \qquad \frac{e_1 \to e_2}{v\ e_1 \to v\ e_2} \qquad \frac{e_1 \to e_2}{e_1\ e \to e_2\ e}$$

**Exercice 2**

1. *Donnez la syntaxe des valeurs $v$ du $\lambda$-calcul sans constante.*
2. *Que signifie « $e\{x \mapsto v\}$ » ? Que vaut « $((\texttt{fun}\ x \Rightarrow xy)\{x \mapsto (\texttt{fun}\ x \Rightarrow x)\})\{y \mapsto x\}$ » ?*
3. *Parmi ces règles de sémantique, quelles sont les règles de réduction ? Quelles sont les règles de passage au contexte ?*
4. *Évaluez les termes suivants du $\lambda$-calcul. Lesquels terminent ? Donnez leurs arbres de preuve.*
   (a) $(\texttt{fun}\ x \Rightarrow x)\ (\texttt{fun}\ x \Rightarrow x)$
   (b) $\texttt{let}\ id\ x = x\ \texttt{in}\ id\ id$
   (c) $(\texttt{fun}\ x \Rightarrow x\,x)\ (\texttt{fun}\ x \Rightarrow x\,x)$
5. *Quels sont les termes bloqués de ce langage ?*

□

## 2.2 Les combinateurs $S$, $K$ et $I$ en appel par valeur

On se donne un nouveau langage, dit "des combinateurs SKI", dont la syntaxe est la suivante :

| | | | |
|---|---|---|---|
| $v$ | $::=$ | | ***Valeurs*** |
| | $\mid$ | $I$ | *Combinateur I* |
| | $\mid$ | $S$ | *Combinateur S* |
| | $\mid$ | $K$ | *Combinateur K* |
| | $\mid$ | $(S\,v)$ | *Application partielle de S* |
| | $\mid$ | $((S\,v)v)$ | |
| | $\mid$ | $(K\,v)$ | *Application partielle de K* |
| $t$ | $::=$ | | ***Termes*** |
| | $\mid$ | $v$ | *Valeur* |
| | $\mid$ | $(t\,t)$ | *Application* |

Dans la suite, on suppose que l'application associe à gauche : on écrira donc « $(t_1\,t_2\,t_3)$ » pour « $((t_1\,t_2\,)t_3)$ ».
Les règles de sémantiques opérationnelles de ce langage sont données par les règles d'inférences suivantes :

$$\frac{t_1 \to t_2}{v\ t_1 \to v\ t_2} \qquad \frac{t_1 \to t_2}{t_1\ t \to t_2\ t}$$

$$\frac{}{(I\,v) \to v} \qquad \frac{}{(K\,v_1\,v_2) \to v_1} \qquad \frac{}{(S\,v_1\,v_2\,v_3) \to v_1\,v_3\,(v_2\,v_3)}$$

**Exercice 3**

1. *Évaluez les termes suivants. Lesquels terminent ? Vous donnerez les arbres de preuve correspondants.*
   (a) $S\,K\,I\,I$
   (b) $S\,K\,K\,(S\,K\,K)$
   (c) $S\,I\,I\,(S\,I\,I)$
2. *Montrez par induction structurelle que :*
$$\forall t, t \equiv v \vee \exists t', t \to t'$$
*En d'autres termes, il n'existe pas de termes bloqués dans ce langage car tout terme $t$ est soit une valeur, soit un terme que l'on peut réduire.*

□

## 2.3 Traduction du $\lambda$-calcul vers les combinateurs $S$, $K$ et $I$, et traduction inverse

On définit les deux fonctions $T$ et $A$ formant la traduction des expressions closes du $\lambda$-calcul vers les termes du langage SKI de la façon suivante :

$$\begin{array}{rcll} T(x) & = & x & \\ T(\texttt{fun}\ x \Rightarrow e) & = & A(x, T(e)) & \\ T(e_1\ e_2) & = & (T(e_1)\,T(e_2)) & \\ & & & \\ A(x,x) & = & I & \\ A(x,y) & = & (K\,y) & \text{si } x \neq y \\ A(x, e_1\ e_2) & = & (A(x,e_1)\,A(x,e_2)) & \end{array}$$

**Exercice 4**

1. *À quelle condition une expression du $\lambda$-calcul est-elle close ? Donnez un exemple d'expression close et un exemple d'expression non close.*
2. *Vérifiez que la traduction est correcte sur les expressions suivantes :*
   (a) $(\texttt{fun}\ x \Rightarrow x)\ (\texttt{fun}\ x \Rightarrow x)$
   (b) $\texttt{let}\ id = (\texttt{fun}\ x \Rightarrow x)\ \texttt{in}\ id\ id$
   (c) $(\texttt{fun}\ x \Rightarrow x\,x)\ (\texttt{fun}\ x \Rightarrow x\,x)$

   *c'est-à-dire, pour chaque expression $e$ :*
   *– calculez $T(e)$ ;*
   *– évaluez le terme $T(e)$ en une valeur $v$ si elle existe ;*
   *– et vérifiez que $v$ est bien la traduction de l'évaluation de l'expression $e$ par les règles du $\lambda$-calcul.*
3. *Proposez une fonction de traduction des termes du langage SKI vers les expressions du $\lambda$-calcul.*
   *(Indication : On commencera par associer une expression raisonnable du $\lambda$-calcul à chacun des combinateurs S, K et I.)*
4. *(Bonus) Comment étendre le langage SKI, la fonction $T$ et la fonction $A$ de façon à obtenir une traduction de l'ensemble des expressions du $\lambda$-calcul (expressions non closes incluses) ?*

□