Contrôle partiel ILP
Revision: 1.2

Christian Queinnec

novembre 2005

Conditions générales

Cet examen est formé de deux exercices en plusieurs questions auxquelles vous pouvez répondre dans l'ordre qui vous plait. L'examen dure 2 heures. Tous les documents sont autorisés.

1 Existence de variables

On souhaite adjoindre au langage ILP2 un nouveau mot clé nommé `exists` permettant de former des expressions testant si une variable est présente ou pas. Ce nouveau dialecte sera baptisé ILP2ex.

On dira qu'une variable locale est présente si le mot clé `exists` est dans la portée de cete variable locale ; on dira qu'une variable globale est présente si elle est lue ou écrite quelque part dans le programme (tester son existence avec `exists` ne lit ni n'écrit la variable concernée). Ainsi, tous les programmes suivants impriment (entre autres) OK.

```
if not (exists x)
then print "OK"
fi
```

```
let x = 1 in
  if exists x
  then print "OK"
  fi
```

```
let x = 1 in
  print(x);
if not (exists x)
then print "OK"
fi
```

```
x = 1;
if exists x
then print "OK"
fi
```

```
if exists x
then print "OK"
fi
print(x);
```

Question 1

Décrire une syntaxe XML pour la représentation de cette nouvelle expression.

Question 2

Écrire une grammaire (le schéma RelaxNG compact) d'ILP2, nommée *grammar2ex.rnc*, correspondant au dialecte ILP2ex.

Question 3

Décrire, en Java (la classe se nommera `fr.upmc.ilp.ilp2ex.CEASTexists.java`), l'AST représentant une expression de mot-clé `exists`.

Question 4

Implanter, pour l'interprète, le mot-clé `exists` pour les seules variables locales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2ex.CEASTexists.java`.

Question 5

Implanter, pour l'interprète, le mot-clé `exists` pour les variables locales et globales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2ex.CEASTexists.java`.

Question 6

Implanter, pour le compilateur vers C, le mot-clé `exists` pour les seules variables locales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2ex.CEASTexists.java`.

Question 7

Implanter, pour le compilateur vers C, le mot-clé `exists` pour les variables locales et globales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2ex.CEASTexists.java`.

2 Définition de variables

On souhaite adjoindre au langage ILP2 un nouveau mot clé nommé `defined` permettant de former des expressions testant si une variable a une valeur ou pas. Ce nouveau dialecte sera baptisé ILP2def.

Une variable locale a toujours une valeur puisque les instructions les introduisant comportent nécessairement une expression pour l'initialiser. Une variable globale n'a de valeur qu'après avoir été initialisée c'est-à-dire avoir été affectée. Les programmes qui suivent impriment tous (entre autres) OK et n'impriment jamais KO.

```
if not (defined x)
then print "OK"
fi
```

```
let x = 1 in
  if defined x
  then print "OK"
  fi
```

```
let x = 1 in
  print(x);
if not (defined x)
then print "OK"
fi
```

```
x = 1;
if defined x
then print "OK"
fi
```

```
if defined x
then print "KO"
fi
x = 1;
if defined x
then print "OK"
fi
```

Question 8

Décrire une syntaxe XML pour la représentation de cette nouvelle expression.

Question 9

Écrire une grammaire (le schéma RelaxNG compact) d'ILP2, nommée *grammar2def.rnc*, correspondant au dialecte ILP2def.

Question 10

Décrire, en Java (la classe se nommera `fr.upmc.ilp.ilp2def.CEASTdefined.java`), l'AST représentant une expression de mot-clé `defined`.

Question 11

Implanter, pour l'interprète, le mot-clé `defined` pour les seules variables locales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2def.CEASTdefined.java`.

Question 12

Implanter, pour l'interprète, le mot-clé `defined` pour les variables locales et globales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2def.CEASTdefined.java`.

Question 13

Implanter, pour le compilateur vers C, le mot-clé `defined` pour les seules variables locales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2def.CEASTdefined.java`.

Question 14

Implanter, pour le compilateur vers C, le mot-clé `defined` pour les variables locales et globales. L'implantation sera fournie sous la forme des méthodes à ajouter à la classe `fr.upmc.ilp.ilp2def.CEASTdefined.java`.