UFR 922 Informatique
Master Informatique – Spécialité STL
Module ILP

TD7 — Intégration

Jacques Malenfant, Christian Queinnec

1 Panorama des nouveautés d'ILP4

Survol des principales nouveautés dans la mise en oeuvre d'ILP4. Lecture de code et réponse aux questions des étudiants.

Points à revoir par rapport au cours 7 :

– Utilisation de la délégation pour réutiliser du code existant dans une classe dont on ne peut hériter.
  – installation du délégué et appel
  – réduction de type (« *narrowing* »)
– Parcours de l'AST en utilisant la réflexion et les annotations pour trouver les accesseurs qui retournent des noeuds ou des tableaux de noeuds d'AST. Décortiquage de la méthode `inline` sur la classe `CEAST`.
– Analyseur fondé sur la réflexion et les méthodes statiques `parse`.

2 Les phases d'analyse

Le programme initial à étudier est :

```
<?xml version='1.0' encoding='ISO-8859-15' ?>
<programme2><!-- test:name description='exo TD7'-->
  <definitionFonction nom='f1'>
    <variables><variable nom='x'/></variables>
    <corps>
      <operationBinaire operateur='-'>
        <operandeGauche><variable nom='x'/></operandeGauche>
        <operandeDroit><entier valeur='1'/></operandeDroit>
      </operationBinaire>
    </corps>
  </definitionFonction>
  <definitionFonction nom='fr2'>
    <variables><variable nom='y'/></variables>
    <corps>
      <alternative>
        <condition>
          <operationBinaire operateur='=='>
            <operandeGauche><variable nom='y'/></operandeGauche>
            <operandeDroit><entier valeur='0'/></operandeDroit>
          </operationBinaire>
        </condition>
        <consequence>
          <variable nom='y'/>
        </consequence>
        <alternant>
          <invocation>
```

```
            <fonction><variable nom='fr2'/></fonction>
            <arguments>
              <invocation>
                <fonction><variable nom='f1'/></fonction>
                <arguments><variable nom='y'/></arguments>
              </invocation>
            </arguments>
          </invocation>
        </alternant>
      </alternative>
    </corps>
  </definitionFonction>
  <invocation>
    <fonction><variable nom='fr2'/></fonction>
    <arguments><entier valeur='5'/></arguments>
  </invocation>
</programme2>
```

Pour les enseignants : on peut à cet endroit suivre les appels de `CEASTParser` pour l'analyse de cet arbre de manière à illustrer le mode de fonctionnement du nouvel analyseur.

L'arbre de syntaxe abstraite après l'analyse syntaxique est :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<fr.upmc.ilp.ilp4.ast.CEASTprogram>
<functionDefinitions>
  <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1000" name="f1" recursive="false">
    <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1001" mangledName="f1_4" name="f1"/>
    <invokedFunctions/>
    <variables>
      <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1002" mangledName="x_1" name="x"/>
    </variables>
    <body>
      <fr.upmc.ilp.ilp4.ast.CEASTsequence id="1003">
        <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1004" operation="-">
          <fr.upmc.ilp.ilp4.ast.CEASTreference id="1005">
            <variable>
              <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1006" mangledName="x_2" name="x"/>
            </variable>
          </fr.upmc.ilp.ilp4.ast.CEASTreference>
          <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1007" value="1"/>
        </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
      </fr.upmc.ilp.ilp4.ast.CEASTsequence>
    </body>
  </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
  <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1008" name="fr2" recursive="false">
    <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1009" mangledName="fr2_12" name="fr2"/>
    <invokedFunctions/>
    <variables>
      <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1010" mangledName="y_5" name="y"/>
    </variables>
    <body>
      <fr.upmc.ilp.ilp4.ast.CEASTsequence id="1011">
        <fr.upmc.ilp.ilp4.ast.CEASTalternative id="1012" ternary="true">
          <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1013" operation="==">
            <fr.upmc.ilp.ilp4.ast.CEASTreference id="1014">
              <variable>
                <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1015" mangledName="y_6" name="y"/>
              </variable>
            </fr.upmc.ilp.ilp4.ast.CEASTreference>
            <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1016" value="0"/>
          </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
          <fr.upmc.ilp.ilp4.ast.CEASTsequence id="1017">
            <fr.upmc.ilp.ilp4.ast.CEASTreference id="1018">
              <variable>
                <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1019" mangledName="y_7" name="y"/>
              </variable>
```

```
            </fr.upmc.ilp.ilp4.ast.CEASTreference>
          </fr.upmc.ilp.ilp4.ast.CEASTsequence>
          <fr.upmc.ilp.ilp4.ast.CEASTsequence id="1020">
            <fr.upmc.ilp.ilp4.ast.CEASTinvocation id="1021">
              <function>
                <fr.upmc.ilp.ilp4.ast.CEASTreference id="1022">
                  <variable>
                    <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1023" mangledName="fr2_8" name="fr2"/>
                  </variable>
                </fr.upmc.ilp.ilp4.ast.CEASTreference>
              </function>
              <arguments>
                <fr.upmc.ilp.ilp4.ast.CEASTinvocation id="1024">
                  <function>
                    <fr.upmc.ilp.ilp4.ast.CEASTreference id="1025">
                      <variable>
                        <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1026" mangledName="f1_9" name="f1"/>
                      </variable>
                    </fr.upmc.ilp.ilp4.ast.CEASTreference>
                  </function>
                  <arguments>
                    <fr.upmc.ilp.ilp4.ast.CEASTreference id="1027">
                      <variable>
                        <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1028" mangledName="y_10" name="y"/>
                      </variable>
                    </fr.upmc.ilp.ilp4.ast.CEASTreference>
                  </arguments>
                </fr.upmc.ilp.ilp4.ast.CEASTinvocation>
              </arguments>
            </fr.upmc.ilp.ilp4.ast.CEASTinvocation>
          </fr.upmc.ilp.ilp4.ast.CEASTsequence>
        </fr.upmc.ilp.ilp4.ast.CEASTalternative>
      </fr.upmc.ilp.ilp4.ast.CEASTsequence>
    </body>
  </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
  </functionDefinitions>
  <globalVariables/>
  <programBody>
    <fr.upmc.ilp.ilp4.ast.CEASTsequence id="1029">
      <fr.upmc.ilp.ilp4.ast.CEASTinvocation id="1030">
        <function>
          <fr.upmc.ilp.ilp4.ast.CEASTreference id="1031">
            <variable>
              <fr.upmc.ilp.ilp4.ast.CEASTvariable id="1032" mangledName="fr2_13" name="fr2"/>
            </variable>
          </fr.upmc.ilp.ilp4.ast.CEASTreference>
        </function>
        <arguments>
          <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1033" value="5"/>
        </arguments>
      </fr.upmc.ilp.ilp4.ast.CEASTinvocation>
    </fr.upmc.ilp.ilp4.ast.CEASTsequence>
  </programBody>
</fr.upmc.ilp.ilp4.ast.CEASTprogram>
```

2.1 Normalisation

La normalisation réalise :

1. l'unification des noeuds variables,
2. leur classification en :
   - `CEASTlocalVariable`
   - `CEASTglobalVariable`

– `CEASTglobalFunctionVariable`
– `CEASTpredefinedVariable`
de même que les affectations et les invocations, et

3. la mise sous une fonction globale du corps du programme.

Quelles sont le noeuds variables et comment seront-ils classifiés ?

Rép :

– le noeud variable `f1` qui est le nom de la première fonction ainsi que la référence à ce nom de fonction dans l'invocation à l'intérieur de `f2r` sous la forme d'une `CEASTglobalFunctionVariable` ;
– le noeud de la variable `x` paramètre de `f1` et la référence à cette variable dans le corps de `f1` sous la forme d'une `CEASTlocalVariable` ;
– le noeud variable `f2r` nom de la seconde fonction et les références à cette fonction dans le corps de `f2r` et dans le corps du programme, sous la forme d'une `CEASTglobalFunctionVariable` ;
– le noeud dela variabel `y` paramètre de `f2r` ainsi que les références à cette variable dans la condition de l'alternative et dans l'alternant de cette condition dans le corps de `f2r`, sous la forme d'une `CEASTlocalVariable`.

Donnez la forme générale de l'arbre de syntaxe abstraite après la normalisation.

Rép : Après la normalisation l'arbre de syntaxe abstraite prend la forme suivante :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<fr.upmc.ilp.ilp4.ast.CEASTprogram>
  <functionDefinitions>
    <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1034" name="f1" recursive="false">
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1035" mangledName="f1_17" name="f1"/>
      <invokedFunctions/>
      <variables>
        <fr.upmc.ilp.ilp4.ast.CEASTlocalVariable id="1036" mangledName="x_19" name="x"/>
      </variables>
      <body>
        <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1037" operation="-">
          <fr.upmc.ilp.ilp4.ast.CEASTreference id="1038">
            <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1036"/></variable>
          </fr.upmc.ilp.ilp4.ast.CEASTreference>
          <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1039" value="1"/>
        </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
      </body>
    </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
    <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1040" name="fr2" recursive="false">
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1041" mangledName="fr2_18" name="fr2"/>
      <invokedFunctions/>
      <variables>
        <fr.upmc.ilp.ilp4.ast.CEASTlocalVariable id="1042" mangledName="y_20" name="y"/>
      </variables>
      <body>
        <fr.upmc.ilp.ilp4.ast.CEASTalternative id="1043" ternary="true">
          <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1044" operation="==">
            <fr.upmc.ilp.ilp4.ast.CEASTreference id="1045">
              <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1042"/></variable>
            </fr.upmc.ilp.ilp4.ast.CEASTreference>
            <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1046" value="0"/>
          </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
          <fr.upmc.ilp.ilp4.ast.CEASTreference id="1047">
            <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1042"/></variable>
          </fr.upmc.ilp.ilp4.ast.CEASTreference>
          <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1048" inlined="false">
            <globalFunction>
              <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1041"/>
            </globalFunction>
            <arguments>
              <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1049" inlined="false">
                <globalFunction>
```

```
                <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1035"/>
              </globalFunction>
              <arguments>
                <fr.upmc.ilp.ilp4.ast.CEASTreference id="1050">
                  <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1042"/></variable>
                </fr.upmc.ilp.ilp4.ast.CEASTreference>
              </arguments>
            </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
          </arguments>
        </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
      </fr.upmc.ilp.ilp4.ast.CEASTalternative>
    </body>
  </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
  <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1051" name="ilpFUNCTION" recursive="false">
    <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1052" mangledName="ilpFUNCTION_21"
                          name="ilpFUNCTION"/>
    <invokedFunctions/>
    <variables/>
    <body>
      <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1053" inlined="false">
        <globalFunction>
          <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1041"/>
        </globalFunction>
        <arguments>
          <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1054" value="5"/>
        </arguments>
      </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
    </body>
  </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
</functionDefinitions>
<globalVariables/>
<programBody>
  <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1055" inlined="false">
    <globalFunction>
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1052"/>
    </globalFunction>
    <arguments/>
  </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
</programBody>
</fr.upmc.ilp.ilp4.ast.CEASTprogram>
```

2.2 Recherche des fonctions invoquées

La recherche des fonctions invoquées calcule pour chaque fonction f la liste des variables désignant les fonctions qui sont invoquées directement ou indirectement par f.

Quelles sont les listes de fonctions qui sont calculées pour ce programme ?

Rép. : Voici les listes :

`f1` : liste vide.

`f2r` : la variable globale désignant `f2r` et celle désignant `f1`.

corps du programme :

Donnez le contenu de l'arbre de syntaxe abstraite après calcul des fonctions invoquées.

Rép. : Après la recherche des fonctions invoquées, l'abre de synatxe abstraite prend la forme :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<fr.upmc.ilp.ilp4.ast.CEASTprogram>
  <functionDefinitions>
    <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1056" name="f1" recursive="false">
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1057" mangledName="f1_17" name="f1"/>
```

```
  <invokedFunctions/>
  <variables>
    <fr.upmc.ilp.ilp4.ast.CEASTlocalVariable id="1058" mangledName="x_19" name="x"/>
  </variables>
  <body>
    <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1059" operation="-">
      <fr.upmc.ilp.ilp4.ast.CEASTreference id="1060">
        <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1058"/></variable>
      </fr.upmc.ilp.ilp4.ast.CEASTreference>
      <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1061" value="1"/>
    </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
  </body>
</fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
<fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1062" name="fr2" recursive="true">
  <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1063" mangledName="fr2_18" name="fr2"/>
  <invokedFunctions>
    <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1063"/>
    <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1057"/>
  </invokedFunctions>
  <variables>
    <fr.upmc.ilp.ilp4.ast.CEASTlocalVariable id="1064" mangledName="y_20" name="y"/>
  </variables>
  <body>
    <fr.upmc.ilp.ilp4.ast.CEASTalternative id="1065" ternary="true">
      <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1066" operation="==">
        <fr.upmc.ilp.ilp4.ast.CEASTreference id="1067">
          <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1064"/></variable>
        </fr.upmc.ilp.ilp4.ast.CEASTreference>
        <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1068" value="0"/>
      </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
      <fr.upmc.ilp.ilp4.ast.CEASTreference id="1069">
        <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1064"/></variable>
      </fr.upmc.ilp.ilp4.ast.CEASTreference>
      <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1070" inlined="false">
        <globalFunction>
          <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1063"/>
        </globalFunction>
        <arguments>
          <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1071" inlined="false">
            <globalFunction>
              <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1057"/>
            </globalFunction>
            <arguments>
              <fr.upmc.ilp.ilp4.ast.CEASTreference id="1072">
                <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1064"/></variable>
              </fr.upmc.ilp.ilp4.ast.CEASTreference>
            </arguments>
          </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
        </arguments>
      </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
    </fr.upmc.ilp.ilp4.ast.CEASTalternative>
  </body>
</fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
<fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1073" name="ilpFUNCTION" recursive="false">
  <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1074" mangledName="ilpFUNCTION_21"
                name="ilpFUNCTION"/>
    <invokedFunctions>
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1063"/>
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1057"/>
    </invokedFunctions>
    <variables/>
    <body>
      <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1075" inlined="false">
        <globalFunction>
          <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1063"/>
        </globalFunction>
        <arguments>
```

```
            <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1076" value="5"/>
          </arguments>
        </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
      </body>
    </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
  </functionDefinitions>
  <globalVariables/>
  <programBody>
    <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1077" inlined="false">
      <globalFunction>
        <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1074"/>
      </globalFunction>
      <arguments/>
    </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
  </programBody>
</fr.upmc.ilp.ilp4.ast.CEASTprogram>
```

2.3 Intégration

L'intégration calcule pour chaque invocation de fonctions globales non-récursives une forme syntaxique représentant l'expression intégrée pouvant remplacer l'invocation.

Quelles sont les invocations qui vont être intégrées dans ce programme ?

Rép. : Les invocations intégrées sont :

– l'appel à f1 dans le corps de `f2r`, et
– l'appel à la fonction globale introduite pour le corps du programme dans le nouveau corps du programme.

Quelle est la forme prise par ces deux intégrations (en syntaxe caméllienne) ?

Rép. :

Pour l'appel à `f1` :

```
let x36 = y37     % y37
in x36 - 1
```

Pour l'appel à la fonction programme :

```
let          % fonction sans paramètre, donc pas de variable locale
in f2r(5)
```

Donnez la forme de l'arbre de syntaxe abstraite après intégration.

Rép. : Après intégration, l'AST prend la forme :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<fr.upmc.ilp.ilp4.ast.CEASTprogram>
  <functionDefinitions>
    <fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition id="1078" name="fr2" recursive="true">
      <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1079" mangledName="fr2_18" name="fr2"/>
      <invokedFunctions>
        <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1079"/>
        <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1080" mangledName="f1_17" name="f1"/>
      </invokedFunctions>
      <variables>
        <fr.upmc.ilp.ilp4.ast.CEASTlocalVariable id="1081" mangledName="y_20" name="y"/>
      </variables>
      <body>
        <fr.upmc.ilp.ilp4.ast.CEASTalternative id="1082" ternary="true">
          <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1083" operation="==">
```

```
          <fr.upmc.ilp.ilp4.ast.CEASTreference id="1084">
            <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1081"/></variable>
          </fr.upmc.ilp.ilp4.ast.CEASTreference>
          <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1085" value="0"/>
        </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
        <fr.upmc.ilp.ilp4.ast.CEASTreference id="1086">
          <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1081"/></variable>
        </fr.upmc.ilp.ilp4.ast.CEASTreference>
        <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1087" inlined="false">
          <globalFunction>
            <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1079"/>
          </globalFunction>
          <arguments>
            <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1088" inlined="true">
              <inlined>
                <fr.upmc.ilp.ilp4.ast.CEASTlocalBlock id="1089">
                  <variables>
                    <fr.upmc.ilp.ilp4.ast.CEASTlocalVariable id="1090" mangledName="x_19" name="x"/>
                  </variables>
                  <initialisations>
                    <fr.upmc.ilp.ilp4.ast.CEASTreference id="1091">
                      <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1081"/></variable>
                    </fr.upmc.ilp.ilp4.ast.CEASTreference>
                  </initialisations>
                  <body>
                    <fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation id="1092" operation="-">
                      <fr.upmc.ilp.ilp4.ast.CEASTreference id="1093">
                        <variable><fr.upmc.ilp.ilp4.ast.CEASTlocalVariable idref="1090"/></variable>
                      </fr.upmc.ilp.ilp4.ast.CEASTreference>
                      <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1094" value="1"/>
                    </fr.upmc.ilp.ilp4.ast.CEASTbinaryOperation>
                  </body>
                </fr.upmc.ilp.ilp4.ast.CEASTlocalBlock>
              </inlined>
              <globalFunction>
                <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1080"/>
              </globalFunction>
              <arguments>
                <fr.upmc.ilp.ilp4.ast.CEASTreference idref="1091"/>
              </arguments>
            </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
          </arguments>
        </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
      </fr.upmc.ilp.ilp4.ast.CEASTalternative>
    </body>
    </fr.upmc.ilp.ilp4.ast.CEASTfunctionDefinition>
  </functionDefinitions>
  <globalVariables/>
  <programBody>
    <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1095" inlined="true">
      <inlined>
        <fr.upmc.ilp.ilp4.ast.CEASTlocalBlock id="1096">
          <variables/>
          <initialisations/>
          <body>
            <fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation id="1097" inlined="false">
              <globalFunction>
                <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable idref="1079"/>
              </globalFunction>
              <arguments>
                <fr.upmc.ilp.ilp4.ast.CEASTinteger id="1098" value="5"/>
              </arguments>
            </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
          </body>
        </fr.upmc.ilp.ilp4.ast.CEASTlocalBlock>
      </inlined>
      <globalFunction>
```

```
        <fr.upmc.ilp.ilp4.ast.CEASTglobalFunctionVariable id="1099" mangledName="ilpFUNCTION_21"
                    name="ilpFUNCTION"/>
      </globalFunction>
      <arguments/>
    </fr.upmc.ilp.ilp4.ast.CEASTglobalInvocation>
  </programBody>
</fr.upmc.ilp.ilp4.ast.CEASTprogram>
```

3 Exercices

3.1 Ajout de l'annotation `inline` à ILP4

Ajouter l'annotation `inline` à ILP4, c'est-à-dire une indication à mettre sur l'invocation des fonctions pour indiquer ceux qui doivent être intégrées. Écrire une extension à ILP4 qui restreint l'intégration aux seules fonctions annotées.

3.2 Dépliage fini des fonctions directement récursives

Modifier votre extension précédente pour admettre des annotations du genre `inline n` sur les appels à des fonctions récursives indiquant qu'on souhaite un dépliage de niveau n (correspondant à `n` appels récursifs.

Rép. : Syntaxiquement, il s'agit de permettre d'avoir un entier comme valeur de l'attribut `inline`. Cet entier doit ensuite être utilisé pour arrêter le niveau d'intégration récursive. L'idée est de calculer l'intégration en posant comme valeur de l'attribut `inlined` l'appel à la fonction, puis en itérant n fois pour remplacer la valeur de cet attribut par une version ou l'appel récursif est intégrer en utilisant la valeur d'intégration de l'itération précédente.