Compilation – TP 6 :
Analyse de vivacité en RETROLIX

Université Paris Diderot – Master 1

(2016-2017)

On rappelle que vous devez faire des *commits* réguliers (à chaque modification de votre code) pour que nous puissions suivre votre avancement.

L'objectif de ce TP est d'implémenter l'analyse de vivacité des variables vue en cours et définie par les équations suivantes :

$$\begin{array}{rcl} \mathrm{in}(n) & = & \mathrm{use}(n) \cup (\mathrm{out}(n) \setminus \mathrm{def}(n)) \\ \mathrm{out}(n) & = & \bigcup_{s \in successors(n)} \mathrm{in}(s) \end{array}$$

Exercice 1 *Analyse de vivacité*

1. *À quoi correspond le n dans les équations précédentes ?*
2. *Que représentent les ensembles $use(n)$ et $def(n)$?*
3. *Quel est le type des valeurs produites par in et out ?*
4. *Comment résoudre un tel système d'équations récursives ?*
5. *Déterminez in et out pour le programme suivant.*

```
  a <- 0
  c <- 0
L:
  b <- a + 1
  c <- c + b
  a <- b + 2
  if c > 42 then L
  ret c
```

6. *Comparez l'exécution de votre algorithme en fonction du choix du parcours du programme. L'algorithme converge-t-il plus vite en descendant ou en remontant dans le graphe de flot de controle ?*
7. *Pourquoi l'usage d'une "file de travail" est utile pour optimiser cet algorithme ?*
8. *Complétez la fonction* `RegisterAllocation.def`.
9. *Complétez la fonction* `RegisterAllocation.use`.
10. *Complétez la fonction* `RegisterAllocation.predecessors`.
11. *Complétez la fonction* `RegisterAllocation.liveness_analysis`.

□