TP2

Utilisez le projet fourni dans tp2.tar.gz, comme pour le TP1 (importez-le sous Eclipse, configurez l'exécution de MyTest.java pour qu'il reçoive un fichier en entrée).

Ce TP comporte 3 parties et s'appuie sur le CM3. Chaque partie correspond à un fichier à compléter. Il y a donc 3 fichiers à rendre au final. La quatrième partie, optionnelle, correspond à un quatrième fichier.

Partie 1 : analyse de durée de vie (fichier TP2Liveness.java)

Complétez le fichier pour construire une analyse de durée de vie. Une itération de l'analyse correspond à une exécution de la fonction onePass(), qui est donc la fonction que vous devez compléter. Vous devez aussi préciser la condition d'arrêt dans isFixedPoint().

Lancez MyTest pour voir les différentes étapes du calcul de point fixe de l'analyse obtenue (affichage de liveIn et liveOut).

Partie 2 : graphe d'interférence (fichier TP2InterferenceGraph.java)

Complétez le fichier correspondant pour que la classe corresponde au graphe d'interférence vu en cours. Remarquer que cette classe hérite d'un graphe non-orienté (la création de nœuds et d'arc ce fait sur le même principe que dans le TP1).

Complétez les fonctions de consultations ident(Node n) et node(Ident id).

Lancez MyTest pour voir le graphe obtenu.

Partie 3 : transformation de programme par fusion de variables (fichier TP2MergeIdent.java)

Cette classe implémente la transformation renommant les variables. Elle hérite de rtl.SimpleTransformation. Vous n'avez qu'une méthode à redéfinir :

```
public Ident transform(Ident id)
```

Cette méthode définie comment transformer chaque variables (sauf les paramètres) d'un programme.
Si vous avez besoin d'initialiser des variables avant l'analyse, faites-le dans le constructeur TP2MergeIdent(...) **pas** dans transform(Program p).

Partie 4 [BONUS] : programmes de tests

Pour cette partie optionnelle, nous vous demandons de rendre, en plus des 3 fichiers .java précédents, quelques fichiers RTL (3 maximums) qui viendront alimenter notre base de tests. Pour chaque fichier, les résultats des exécutions avec l'interpréteur rtl, avant et après transformation seront comparés. Si les résultats diffèrent, nous auront ainsi détecter une erreur dans la transformation. Des points bonus seront attribués aux binômes qui proposent des fichiers RTL particulièrement *performants* (bonne couverture des différents cas tout en étant de taille modeste) pour trouver des erreurs dans les transformations des (autres) étudiants.