Examen final d'ILP

Christian Queinnec

25 janvier 2011

Conditions générales

Cet examen est formé d'un unique problème en plusieurs questions auxquelles vous pouvez répondre dans l'ordre qui vous plait.

Le barème est fixé à 20 ; la durée de l'épreuve est de 3 heures. Tous les documents sont autorisés et notamment ceux du cours.

Votre copie sera formée de fichiers textuels que vous laisserez aux endroits spécifiés dans votre espace de travail pour Eclipse. L'espace de travail pour Eclipse sera obligatoirement nommé `workspace` et devra être un sous-répertoire direct de votre répertoire personnel.

À l'exception des clés USB en lecture seule, tous les appareils électroniques communiquants sont prohibés (et donc notamment les téléphones portables). Vos oreilles ne doivent pas être reliées à ces appareils.

L'examen sera corrigé à la main, il est donc absolument inutile de s'acharner sur un problème de compilation ou sur des méthodes à contenu informatif faible. Il est beaucoup plus important de rendre aisé, voire plaisant, le travail du correcteur et de lui indiquer, par tout moyen à votre convenance, de manière claire, compréhensible et terminologiquement précise, comment vous surmontez cette épreuve. À ce sujet, vos fichiers n'auront que des lignes de moins de 80 caractères, n'utiliseront que le codage ASCII ou UTF-8 enfin, s'abstiendront de tout caractère de tabulation.

Le langage à étendre est ILP4 et vise à produire un simplificateur. Le paquetage Java correspondant à cet examen sera nommé `fr.upmc.ilp.ilp4.simpl` (la dernière lettre est un L minuscule). Sera ramassé, à partir de votre *workspace* (situé sous ce nom directement dans votre HOME), le seul répertoire `ILP/Java/src/fr/upmc/ilp/ilp4/simpl/`

Pour vous éviter de la taper à nouveau, voici l'url du site du master :

```
http://www-master.ufr-info-p6.jussieu.fr/site-annuel-courant/
```

Simplification

Lors d'un examen précédent, il était demandé un simplificateur pour ILP4. Le but de ce nouvel examen est d'étendre ce simplificateur (par héritage). Plusieurs analyses seront demandées :

- simplification des variables locales inutilisées
- simplification des variables inutilisées et non lues
- simplification des expressions inutiles sans effet secondaire
- simplification des expressions constantes

Le code qui vous sera demandé héritera du code donné en correction de l'examen précédent et sera défini, comme mentionné plus haut, dans un paquetage distinct nommé `fr.upmc.ilp.ilp4.simpl`.

Question 1 – Variables inutilisées (3 points)

Une variable locale qui n'est pas utilisée n'est pas utile et peut donc disparaître. Vous définirez précisément la notion d'utilisation (ou plutôt d'inutilisation) d'une variable locale puis vous indiquerez en quoi doit être réécrit un bloc local (unaire ou n-aire) introduisant une variable locale inutile.

Question 2 – Variables inutilisées et non lues (4 points)

Une variable locale ou globale possiblement affectée mais jamais lue n'est pas utile et peut donc disparaître. Vous définirez précisément comment l'on peut déterminer ces variables et comment et en quoi un programme peut être réécrit sans ces variables.

Il peut être utile de définir les prédicats `isRead` (resp. `isWritten`) répondant Vrai si une variable est lue (resp. affectée).

Question 3 – Expressions sans effet secondaire (6 points)

Lorsque la valeur d'une expression est inutile et si elle ne contient aucun effet secondaire alors elle peut éventuellement disparaître. Quels sont les lieux où la valeur d'une expression est inutile ? Quels sont les effets secondaires possibles en ILP ?

Vous indiquerez comment calculer une approximation sûre d'un prédicat, nommé `hasSideEffect`, répondant Vrai si une expression contient un effet secondaire. Ce prédicat peut être approximatif c'est-à-dire répondre aussi Vrai pour des expressions n'ayant pas d'effet secondaire mais trop compliquées à analyser.

Enfin, vous indiquerez comment et en quoi se réécrit un programme comportant des expressions inutiles sans effet secondaire.

Question 4 – Expressions constantes (4 points)

Une expression dont les composants sont connus peut être simplifiée. Ainsi 3*4 peut-il être remplacé par 12. Vous expliquerez comment procéder (comment déterminer les expressions sujettes à cette simplification et en quoi les transformer) à cette simplification (en anglais *constant folding*) partout où elle est possible.

Question 5 – Intégration (3 points)

Pour cette question, nous vous demandons la classe `Process` du paquetage `fr.upmc.ilp.ilp4.simpl` montrant comment vous intégrez les différentes simplifications demandées plus haut.