

TD numéro 3 : Sélection d'instructions

Assembleur – Compilation, ENSIIE

Semestre 3, 2023–24

Exercice 1 : Booléens

1. Proposer une traduction naïve de la condition `not (true and not (x + 4 <> z))`
2. En plus des règles pour `add` vues en cours, on considère le système de réécriture suivant :

$$\text{and}(\text{li}_0, E) \rightarrow \text{li}_0 \quad (1)$$

$$\text{and}(\text{li}_1, E) \rightarrow E \quad (2)$$

$$\text{and}(E, \text{li}_1) \rightarrow E \quad (3)$$

$$\text{and}(E, \text{li}_0) \rightarrow \text{li}_0 \quad (4)$$

$$\text{not}(\text{sne}(E_1, E_2)) \rightarrow \text{seq}(E_1, E_2) \quad (5)$$

$$\text{not}(\text{not}(E)) \rightarrow E \quad (6)$$

Donner la ou les forme(s) normale(s) de la traduction naïve de la question précédente.

3. Le système de réécriture termine-t-il ?
4. Le système de réécriture est-il confluente ? Si non, comment le rendre confluente ?
5. Que penser de la quatrième règle ? Et de la première ?

Exercice 2 : Multiplication

1. Donner une traduction naïve en Untyped Pseudo Pascal de l'expression `4 * (x + 2)`
Pour optimiser la multiplication, on propose la règle suivante :

$$\text{mul}(E, \text{li}_{2^k}) \rightarrow \text{slli}_k(E) \quad (7)$$

(On rappelle que `slli` est l'opération MIPS qui décale les bits contenu dans un registre vers la gauche.) Pour pouvoir traiter également le cas où la constante est à gauche du `mul`, on ajoute également la règle suivante :

$$\text{mul}(\text{li}_n, E) \rightarrow \text{mul}(E, \text{li}_n) \quad (8)$$

2. Donner une forme normale de la traduction obtenue à la question 1.
3. Le système de réécriture est-il terminant ? Justifier.
4. Calculer les paires critiques du système. Lesquelles sont-elles joignables ?
5. Le système de réécriture est-il confluente ? Justifier.