Travaux dirigés et pratiques
TD7 bis - Sémantique naturelle (suite)

Sylvain Chevillard et Damien Pous

21 avril 2008

Au TD précédent, nous avons défini et utilisé une sémantique pour un petit langage impératif simple. Ce TD faisant suite au précédent, commencez par finir le TD précédent.

À l'aide des fonctions définies au TD précédent, nous allons écrire un interpréteur pour notre petit langage impératif et nous allons en prouver la correction.

Nous sommes confronté à un problème désormais bien connu : nous voudrions écrire l'interpréteur sous forme d'une fonction récursive, mais il n'y a pas d'argument structurellement décroissant ; et pour cause, certains programme ne terminent pas !

Nous allons donc écrire un interpréteur partiel, doté d'un argument entier n supplémentaire et tel que, si le programme termine, il existe N tel que pour $n \geq N$, l'interpréteur renvoie le bon résultat. Dans tout autre cas, l'interpréteur renvoie `None`.

Écrire un tel interpréteur `fexec : state -> instr -> nat -> option state.`

Pour montrer la correction de l'interpréteur il faut montrer deux choses : d'une part, si la sémantique naturelle prévoit que le calcul termine, l'interpréteur permet de calculer le résultat pour un n assez grand ; d'autre part, si l'interpréteur retourne un résultat, ce résultat aurait été prédit par la sémantique naturelle. Dans le cadre de ce TD, nous n'allons montrer que la première partie ; libre à vous de vous atteler à la deuxième pour vous entraînez (elle n'est pas plus compliquée).

Pour commencer, nous montrons un lemme qui nous sera utile :

```
Lemma fexec_S : forall n s i s',
  fexec s i n = Some s' -> fexec s i (S n) = Some s'.

Lemma fexec_monotonic : forall n n' s i s',
  (n <= n') -> fexec s i n = Some s' -> fexec s i n' = Some s'.
```

De plus, dans la preuve de correction, nous devrons prendre le maximum de deux nombres. Il n'y a rien qui permette de le faire dans la librairie standard. Nous démontrons donc le lemme suivant :

```
Theorem maximum : forall n1 n2:nat, exists n:nat, n1 <= n /\ n2 <= n.
```

Armés de ces résultats préliminaires, démontrez la première partie de la preuve de correction :

```
Theorem fexec_correct :
  forall s i s', nat_sem s i s' -> exists n:nat, fexec s i n = Some s'.
```

Si vous êtes courageux, faites la deuxième partie (vous énoncerez et démontrerez les lemmes que vous jugerez utiles) :

```
Theorem fexec_correct2 :
  forall n s i s', fexec s i n = Some s' -> sem_nat s i s'.
```