Arithmétique

Preuves assistées par ordinateur – TP 2 – 7 mars 2025

Année scolaire 2024–2025

Exercice 1 : Quelques résultats d'arithmétique

On se propose de redémontrer quelques résultats d'arithmétiques déjà démontrés dans la bibliothèque standard de Coq (on indique entre parenthères le nom des énoncés correspondants).

1. Démontrer que l'addition est commutative (`Nat.add_comm`) et associative (`Nat.add_assoc`) et admet 0 pour neutre (`plus_O_n`).
2. Démontrer que la multiplication est distributive par rapport à l'addition (`Nat.mul_add_distr_l`).
3. Démontrer que la multiplication est commutative (`Nat.mul_comm`) et associative (`Nat.mul_assoc`) et admet 1 pour neutre (`Nat.mul_1_l`) et 0 pour élément absorbant (`Nat.mul_0_l`).

Exercice 2 : Compilation vers une machine à pile

On considère la définition inductive suivante pour représenter les expressions arithmétiques.

```
Inductive expr :=
| Constant (_ : nat)
| Plus (_ _ : expr)
| Times (_ _ : expr).
```

1. Écrire une fonction `eval` qui calcule la valeur de l'expression.

On considère maintenant le langage à pile constitué des listes des commandes suivantes.

```
Inductive command :=
| Push (_ : nat)
| Add
| Mul.
```

2. Écrire une fonction `eval_stack` qui calcule la valeur d'une liste de commandes étant donnéee une pile initiale. On utilisera un type `option` et la valeur `None` en cas d'erreur de pile.
3. Écrire une fonction `compile` qui traduit une expression et une liste de commandes et démontrer que la valeur évaluée est préservée.