Typage de MINIML

Julien FORGET

30 avril 2009

1 Structures et fonctions préliminaires (`types.ml`)

1.1 Types de base

Les types de base de MINIML sont :
- *Entier*
- *Booléen*
- *Fonctionnel*

1. Définissez un type union `type_expr` représentant les types de MINIML.
2. Définissez une fonction `string_of_type:type_expr->string` convertissant un type en chaîne de caractère.

1.2 Environnements de typage

Un environnement sera représenté à l'aide d'une liste d'association (cf le module `List` de la distribution OCAML). Ceci est peu efficace mais simplifie l'implémentation.

1. Pour améliorer la lisibilité, on définit explicitement le type environnement : **`type`** `env=(string*type_expr) list`.
2. Définissez une fonction `get_env_type:env->string->type_expr` retournant le type associé à une variable dans un environnement.
3. Définissez une fonction `add_env_value` qui rajoute une association dans une environnement de typage.

2 Inférence (`typing.ml`)

1. Définissez la fonction `type_of_expr:env->expr->type_expr` calculant le type d'une expression dans une environnement de typage donné. Ne vous souciez pas du polymorphisme. Adaptez les règles d'inférence données en cours.
2. Définissez la fonction de typage d'un programme `type_prog:prog->type_expr list`.

3. Ajoutez le calcul du type du programme analysé dans la fonction `main` définie au cours du premier TP.
4. Affichez le type de chacune des expressions du programme.
5. Mettez à jour le `Makefile`, en ajoutant `types.cmo typing.cmo` juste avant `main.cmo`. Recalculez les dépendances avant de compiler.

3 Tests

Afin de vérifier la correction de votre programme tester avec les expressions suivantes :

- L'expression calculant factorielle 4.
- Une fonction (non récursive et non itérative) retournant la somme des n premiers entiers.
- Une fonction qui retourne 1 si son paramètre est nul, 0 sinon.
- Une fonction retournant le produit de ses deux paramètres.
- Appliquer chacune de ces fonctions sur un exemple correct.
- Appliquer chacune de ces fonctions sur un exemple syntaxiquement correct mais mal typé.
- Effectuer une application partielle si une fonction le permet.

Travail relevé

Votre travail est à envoyer par email à : `julien.forget@onera.fr`.

Vous enverrez à cette adresse l'ensemble des fichiers source (.ml) ainsi que le fichier de test vous ayant permis de vérifier le fonctionnement de votre TP, rassemblés dans une archive (zip ou tar).