Rapport de travail

par Rémi Langevin et Christophe Apollon-Roy

1. Problèmes rencontrés

Trouver le format de base de Llambda

Au départ, nous avions décidé de transformer l'arbre de Sexp en liste. Au début, cela facilitait la tâche. Toutefois, nous avons rapidement eu des difficultés avec les cas plus complexes, dont le lambda. À partir de ce moment, nous avons décidé d'y aller avec du pattern matching directement sur la structure arborescente des S-expressions et ainsi déterminer les patterns associés à chaque type de Sexp.

Gérer les Llet imbriqués et avec assignation multiples

Trouver les patterns associés et comment extraire de multiples déclarations dans un Llet fut une tâche complexe. Nous avons eu besoin de dessiner quelques arbres afin de bien généraliser.

Currying

Le currying nous a couté beaucoup de temps et de modifications dans le code. À un certain point, d'un côté, nous pouvions faire fonctionner les fonctions non-curried, de l'autre les fonctions curried, mais jamais les deux en même temps. Cela se jouait au niveau du Lapp. Sans currying, nous avions:

```
s2l (Scons Snil a) =
   case (s2l a) of
   ...
   (Lapp x y) -> Lapp x y
```

et avec currying:

```
s2l (Scons Snil a) =
   case (s2l a) of
   ...
   (Lapp x y) -> Lapp (Lapp x y) []
```

Nous avions donc eu des difficultés à trouver une méthode adéquate pour les deux cas.

Unsweetner

Se débarasser du sucre syntaxique fut une autre fonction complexe à implémenter sans briser le code déjà existant.

2. Surprises

Nous avons été surpris à quel point les cases, cons et if furent facile à implémenter. Le fait de pouvoir utiliser la structure de l'arbre directement avec du pattern matching fut une autre plaisante surprise.

De plus, malgré la difficulté du travail, nous nous sommes surpris à apprécier l'exercice, bien que nous ne le referions pas une deuxième fois.

Ensuite, nous avions été surpris par la facilité avec laquelle nous pouvions ajouter d'autres fonctions utilisant des fonctions déjà présentes sans se soucier de leurs interactions lorsque celles-ci sont implémentées correctement.

3. Choix

Nous avons choisi de ne pas avoir de souplesse dans la syntaxe acceptée, car trouver les patterns associés à certains modèles devenait une tâche trop coriace.

Aussi, nous avons choisi, en fin de travail, de ne faire aucune vraie différence entre les dlet et les llet dans eval (s2l fait la différence), donc au final, il n'y a aucune différence entre les deux.

4. Options sciemment rejetées

Nous avons malheureusment rejeté le dernier test dans exemples.psil, car nous n'avons pas trouvé de solution à temps pour le résoudre.