Compte rendu TP Compilation n° 1
Analyse lexicale et crible

Nargeot Guillaume
Angelique Lainé

17 octobre 2005

Table des matières

1 Explication du code

Fonction *fct_ transition* : la fonction de transition de l'automate
- 2 paramètres en entrée : l'état courant et le caractère lu
- retourne l'état suivant dans l'automate après la transition

Fonction *get_ token* : représente le scanner incrémental qui rend un lexème à la demande
- 2 paramètres en entrée : la description de l'automate (représenté par l'état de départ, la fonction de transition et la liste des états finaux) et le flot à analyser
- retourne un couple (unité lexicale, lexème) à la fois

Cette fonction fait appel à la fonction *get_ token_ aux*.

Fonction *get_ token_ aux* :
- 3 paramètres en entrée : la description de l'automate, (représenté par l'état de départ, la fonction de transition et la liste des états finaux), le lexème que l'on construit et le flot à analyser
- retourne le lexème détecté

Algorithme de la fonction

On prend le caractère courant du flot
Si on est pas à la fin du flot
alors
    on fait appel à la fonction de transition.
    si la fonction de transition retourne l'etat final E3
    alors
        (le dernier caractère lu ne doit pas être consommé)
        on retourne le lexème trouvé
    sinon
        si la fonction de transition retourne l'état initial
        alors
            (cas où on a lu des commentaires, un espace, une tabulation ou un retour à la ligne)
            on consomme le caractère lu.
            on relance la recherche d'un lexème sans tenir compte du commentaire ou espace ou tabulation ou retour à la ligne trouvé.
        sinon
            si la fonction de transition retourne l'état final E8
            alors
                (le dernier caractère lu doit être consommé)
                on consomme le caractère lu.
                on retourne le lexème trouvé.
            sinon
                on continue récursivement la recherche du lexème.
sinon
    (on est à la fin du flot)
    on retourne le lexème trouvé.

La valeur *unite_lex* correspond au résultat du crible : on compare le lexème trouvé avec les chaînes connues pour retourner le bon couple (unité lexicale, lexème).

Fonction *scanner* : retourne la liste des lexèmes générés à partir du flot d'entrée
- 2 paramètres en entrée : la description de l'automate (représenté par l'état de départ, la fonction de transition et la liste des états finaux) et le flot à analyser
- retourne la liste des couples (unité lexicale, lexème)

2 Questions

Quel est l'intérêt d'avoir un crible séparé dans l'analyseur lexical ?

Séparer le crible dans l'analyseur lexical permet d'avoir un analyseur lexical plus générique. La sous-fonction effectuant crible — *unit_lex* — pourra être changée ou modifiée simplement puisque toutes les unités lexicales sont réunies au sein de la même sous-fonction. La maintenance de cet analyseur lexical en est donc facilitée.

Quels sont les intérêts d'utiliser les types énumérés de CAML ?

L'utilisation de types énumérés en CAML permet de précisément définir les types des paramètres utilisés. Cela permet d'être plus explicite dans les valeurs retournées

et d'éviter les erreurs de typage. Le code est plus optimisé, plus performant.

Quel est l'intérêt d'un scanner incrémental qui rend un lexème à la demande (fonction *get_token*), par rapport à un scanner qui rend la liste des lexèmes du programme ?

L'intérêt de faire un scanner ne rendant qu'un lexème à la fois permet de simplifier l'implémentation de l'analyseur lexical. Le scanner rendant la liste des unités lexicales est développé en faisant appel autant de fois que nécessaire au scanner incrémental. De plus, la gestion et la récupération des erreurs en est simplifiée.

Pourquoi ident est-il considéré comme un terminal et non pas un non-terminal ?

Si Ident était considéré comme un non-terminal, cela signifierait qu'il posséderait un sens sémantique autre qu'identificateur. Or ici, nous devons le détecter comme une unité lexicale, c'est pourquoi Ident est un terminal.

3 Le diagramme de transition