Traduction des langages

Arbre syntaxique abstrait, table des symboles et résolution des identifiants

Objectif :
- Définir l'AST (Abstract Syntax Tree) pour le langage RAT
- Comprendre le traitement réalisé par la passe de résolution des identifiants
- Comprendre l'intérêt de la table des symboles
- Définir la nouvelle structure d'arbre obtenue après la passe de résolution des identifiants
- Définir les actions à réaliser par la passe de résolution des identifiants

1 Le langage Rat

1.1 Grammaire du langage RAT

1. $PROG' \rightarrow PROG\ \$$
2. $PROG \rightarrow FUN\ PROG$
3. $FUN \rightarrow TYPE\ id\ (\ DP\ )\ BLOC$
4. $PROG \rightarrow\ id\ \ BLOC$
5. $BLOC \rightarrow\ \{\ IS\ \}$
6. $IS \rightarrow\ I\ \ IS$
7. $IS \rightarrow \Lambda$
8. $I \rightarrow\ TYPE\ \ id\ =\ E\ ;$
9. $I \rightarrow\ id\ =\ E\ ;$
10. $I \rightarrow\ const\ \ id\ =\ entier\ ;$
11. $I \rightarrow\ print\ \ E;$
12. $I \rightarrow\ if\ \ E\ \ BLOC\ \ else\ \ BLOC$
13. $I \rightarrow\ while\ \ E\ \ BLOC$
14. $I \rightarrow\ return\ \ E\ ;$
15. $DP \rightarrow \Lambda$
16. $DP \rightarrow TYPE\ id\ DP2$
17. $DP2 \rightarrow\ ,\ TYPE\ id\ DP2$
18. $DP2 \rightarrow \Lambda$
19. $TYPE \rightarrow\ bool$
20. $TYPE \rightarrow\ int$
21. $TYPE \rightarrow\ rat$
22. $E \rightarrow id\ (\ CP\ )$
23. $E \rightarrow\ [\ E\ /\ E\ ]$
24. $E \rightarrow\ num\ \ E$
25. $E \rightarrow\ denom\ \ E$
26. $E \rightarrow\ id$
27. $E \rightarrow\ true$
28. $E \rightarrow\ false$
29. $E \rightarrow\ entier$
30. $E \rightarrow\ (\ E\ +\ \ E\ )$
31. $E \rightarrow\ (\ E\ *\ \ E\ )$
32. $E \rightarrow\ (\ E\ =\ \ E\ )$
33. $E \rightarrow\ (\ E\ <\ \ E\ )$
34. $E \rightarrow\ (\ E\ )$
35. $CP \rightarrow\ \Lambda$
36. $CP \rightarrow\ E\ CP2$
37. $CP2 \rightarrow\ ,\ E\ CP2$
38. $CP2 \rightarrow\ \Lambda$

1.2 Exemple

```
bool less (rat a, rat b) {
  return ((num a * denom b ) < ( num b * denom a ));
}

prog{
  rat a = [3/4];
  rat b = [4/5];
  const n = 5;
  int i = 0;
  while(i<n){
    a=(a+a);
    b=(b*b);
    i=(i+1);
  }
  if (less(a, b)) {print a;} else {print b;}
}
```

2 Structure de l'AST pour RAT

▷ **Exercice 1** *Définir la structure de l'AST pour RAT.*

3 Passe de résolution des identifiants

Nous rappelons qu'un compilateur fonctionne par passes, chacune d'elle réalisant un traitement particulier (gestion des identifiants, typage, placement mémoire, génération de code,...). Chaque passe parcourt, et potentiellement modifie, l'AST.

La première passe est une passe de résolution des identifiants. C'est elle qui vérifie la bonne utilisation des identifiants. En particulier, dans le cas du langage RAT, c'est elle qui vérifie que :

- déclaration : vérifier l'absence de double déclaration dans le même bloc ;
- utilisation d'un symbole : vérifier l'existence de ce symbole dans ce bloc ou un bloc englobant ;
- utilisation d'un symbole : vérifier que sa catégorie (variable, constante, fonction) est correcte.

Pour réaliser cette passe, nous avons besoin d'une table des symboles : structure de données associant aux identifiants leurs informations.

3.1 Table des symboles

L'interface du module Tds utilisé en TP est la suivante :

```
(* Définition du type des informations associées aux identifiants *)
type info =
  (* Information associée à une constante : son nom (non indispensable mais aide au test et debbugage) et sa valeur *)
  | InfoConst of string * int
  (* Information associée à une variable : son nom (non indispensable mais aide au test et debbugage),
  son type, et son adresse ie son déplacement (int) par rapport à un registre (string) *)
  | InfoVar of string * typ * int * string
  (* Information associée à une fonction : son nom (utile pour l'appel), son type de retour
```

```
    et la liste des types des paramètres *)
  | InfoFun of string * typ * typ list

(* Table des symboles *)
type tds

(* Données stockées dans la tds et dans les AST : pointeur sur une information *)
type info_ast

(* Création d'une table des symboles à la racine *)
val creerTDSMere : unit -> tds

(* Création d'une table des symboles fille *)
(* Le paramètre est la table mère *)
val creerTDSFille : tds -> tds

(* Ajoute une information dans la table des symboles locale *)
(* tds : la tds courante *)
(* string : le nom de l'identificateur *)
(* info : l'information à associer à l'identificateur *)
(* Si l'identificateur est déjà présent dans TDS, l'information est écrasée *)
(* retour : unit *)
val ajouter : tds -> string -> info_ast -> unit

(* Recherche les informations d'un identificateur dans la tds locale *)
(* Ne cherche que dans la tds de plus bas niveau *)
val chercherLocalement : tds -> string -> info_ast option

(* Recherche les informations d'un identificateur dans la tds globale *)
(* Si l'identificateur n'est pas présent dans la tds de plus bas niveau la recherche est effectuée *)
(* dans sa table mère et ainsi de suite jusqu'à trouver (ou pas) l'identificateur *)
val chercherGlobalement : tds -> string -> info_ast option

(* Affiche la tds locale *)
val afficher_locale : tds -> unit

(* Affiche la tds locale et récursivement *)
val afficher_globale : tds -> unit

(* Créer une information à associer à l'AST àpartir d'une info *)
val info_to_info_ast : info -> info_ast

(* Récupère l'information associée à un noeud *)
val info_ast_to_info : info_ast -> info

(* Modifie le type si c'est une InfoVar, ne fait rien sinon *)
val modifier_type_variable : typ -> info_ast -> unit

(* Modifie les types de retour et des paramètres si c'est une InfoFun, ne fait rien sinon *)
val modifier_type_fonction : typ -> typ list -> info_ast -> unit

(* Modifie l'emplacement (dépl, registre) si c'est une InfoVar, ne fait rien sinon *)
val modifier_adresse_variable : int -> string -> info_ast -> unit
```

3.2 Structure de l'AST post passe de résolution des identifiants

La passe de résolution des identifiants réalise des vérifications mais prépare également les passes suivantes. Elles auront besoin d'avoir accès aux informations des identifiants, il faut donc modifier l'AST de façon à rendre ces informations accessibles.

▷ **Exercice 2** *Définir la structure de l'AST post passe de résolution des identifiants.*

3.3 Actions à réaliser lors de la passe de résolution des identifiants

▷ **Exercice 3** *Définir les actions à réaliser lors de la passe de résolution des identifiants.*