LOG6302 : Quiz 1

Troclet Philippe *1815208*

30 janvier 2017

1 Application des techniques de réingénérie

Une première application serait le calcul de métrique à partir d'une base de code. Une second application serait la génération de diagramme UML. Ces applications nécessitent de construire l'arbre de parsage puis de le parcourir. On remarque qu'on pourrait également réaliser ces applications via des traductions dirigées par la syntaxe.

2 Étude d'une grammaire

Question a

Les non-terminaux de cette grammaire sont *bexpr*,*bterm* et *bfactor*. Les terminaux quant à eux sont **or**, **and**, **not**,**(** et **(**. Enfin le symbole *Start* est *bexpr*. En effet, les non-terminaux sont l'ensemble des symboles que l'on peut retrouver en partie gauche d'une règle. Les terminaux sont alors l'ensemble des symboles qui ne sont pas des non-terminaux. Enfin, les règles de production sont les suivantes :

1. *bexpr* → *bexpr* **or** *bexpr*
2. *bexpr* → *bterm*
3. *bterm* → *bterm* **and** *bterm*
4. *bterm* → *bfactor*
5. *bfactor* → **not** *bfactor*
6. *bfactor* → **(***bexpr***)**
7. *bfactor* → **true**
8. *bfactor* → **false**

Question b

D'après la question précédente, il y a 8 règles de production.

Question c

Le *xor* et le ne font pas partie de la grammaire aussi les deux derniers ne peuvent appartenir au langage. En conséquence, seul **not true and false** fait partie du langage.

Question d

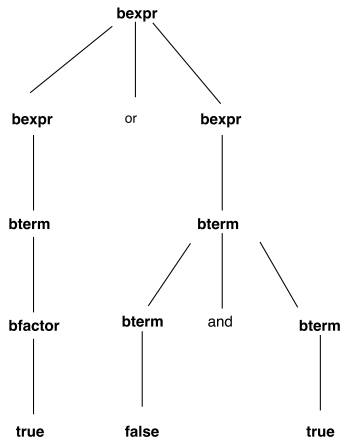

FIGURE 1 – Arbre de syntaxe pour **true or false and true**

Les deux arbres de parsage sont identiques, on peut voir l'arbre correspondant sur la figure 1.

Question e

La grammaire considérée ici est ambiguë, on peut pour cela étudier la chaîne **true or false or true**. En effet, selon qu'on suive une dérivation à gauche ou à droite, on obtient un arbre de parsage différent.

3 Modification d'une grammaire

Afin d'introduire un **xor**, nous proposons la grammaire suivante :

1. *bexpr* → *bexpr* **or** *bexpr*
2. *bexpr* → *exprxor*
3. *exprxor* → *exprxor* **xor** *exprxor*
4. *exprxor* → *bterm*
5. *bterm* → *bterm* **and** *bterm*
6. *bterm* → *bfactor*
7. *bfactor* → **not** *bfactor*
8. *bfactor* → (*bexpr*)
9. *bfactor* → **true**
10. *bfactor* → **false**

4 Création d'une grammaire

Afin de pouvoir reconnaître les numéros de téléphone canadiens, nous proposons la grammaire suivante :

— $S \rightarrow (+1)num|num$
— $num \rightarrow CCC\ CCC\ CCCC$
— $C \rightarrow 0|1|2|3|4|5|6|7|8|9$