In406 projet partie 2

Emeric Gouy

1

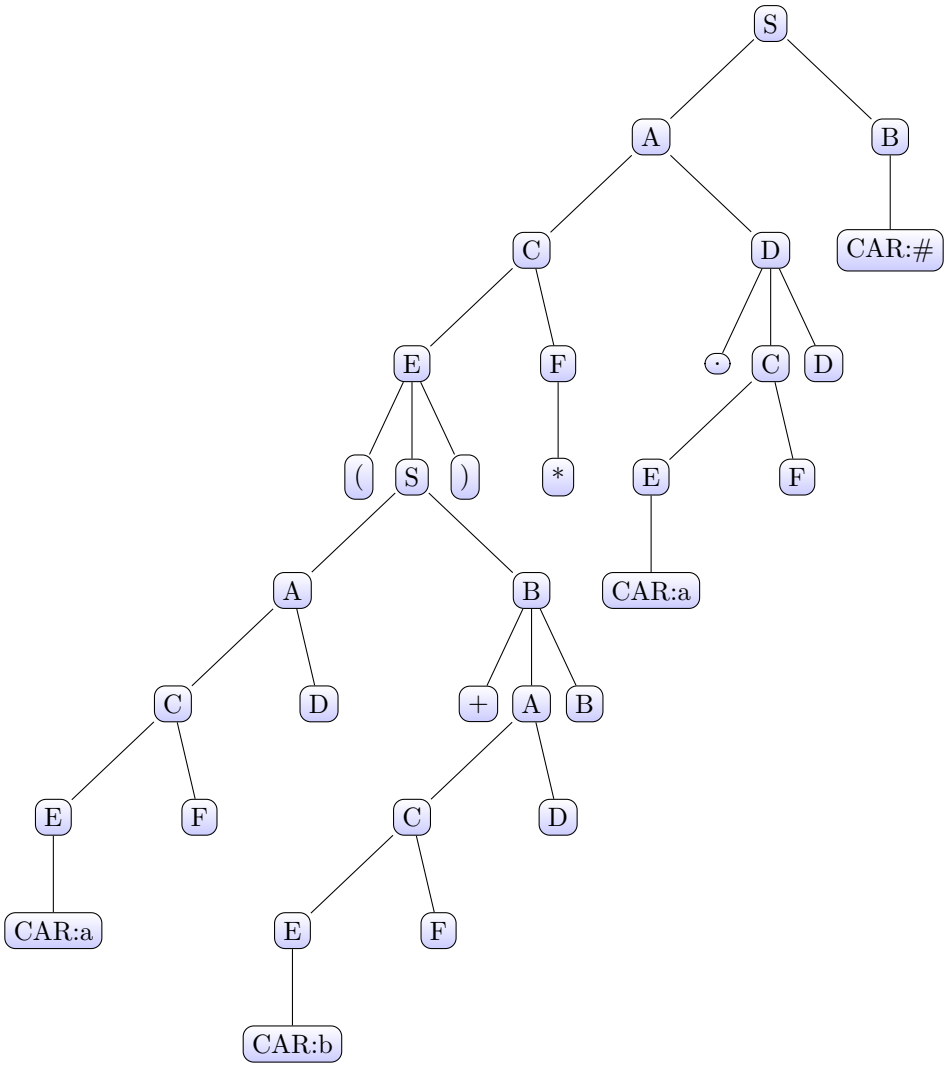

Figure 1: Arbre de dérivation de $(a+b)^*.a\#$

2

L'arbre de dérivation représente les opérations qui ont lieu dans la pile de reconnaissance : en effet, les fils d'un noeud correspondent aux états qu'on empile après avoir dépilé un état et comparé avec le caractère courant sur l'expression régulière. On peut donc imaginer une construction de l'arbre en profondeur : on part de l'état initial S (racine de l'arbre), on le dépile, puis on ajoute les états calculés via la table de transition comme fils à S. On rappelle ensuite cette procédure récursivement sur les fils (en commençant par celui en sommet de pile).

Quand on empile un terminal, ou qu'on a affaire à une transition de la forme $X \to \varepsilon$, la construction s'arrête sur la branche en cours. Si une comparaison avec un terminal est invalide ou qu'il n'y a aucune transition possible à partir du non terminal en cours, la construction s'arrête (en libérant l'arbre qui était en construction) et on renvoie une erreur.

3

On remarque que certains états représentent la présence d'opérateurs dans l'expression régulière traitée : par exemple, B correspond au $+$, D au $.$, F à $*$. Cette observation permet de créer l'arbre de l'expression régulière (exemple d'exécution en figure 2) : on part du sommet de l'arbre de dérivation, et on regarde s'il possède un fils B,D,F qui ait un état témoignant d'une opération en fils. Si oui on crée un noeud correspondant à l'opération dans l'arbre de l'e.r., on appelle récursivement cette procédure sur le fils en question (pour vérifier s'il n'y a pas plusieurs concaténations/additions à la suite par exemple), on attache l'arbre obtenu par récursion au noeud créé. Enfin on appelle récursivement la procédure sur le fils A,C,E qu'on a pas encore traité, et on attache le résultat au noeud de l'opération. Sinon, on rappelle la procédure récursivement sur le fils A,C,E ne correspondant pas à une opération directement. Quand on atteint un caractère, la construction s'arrête dans la branche en cours.

4

On peut reprendre sensiblement la même idée qu'en question 3, sauf que sans l'arbre de dérivation, on ne peut pas regarder si un non-terminal $B/D/F$ correspond à une opération valide en 1er, puisqu'on n'a pas terminé la reconnaissance : il faut s'y prendre un peu autrement, en partant d'abord des feuilles de l'arbre de l'e.r. Quand on empile un caractère, on se crée un arbre à 1 noeud contenant ce caractère. Lorsqu'on empile des non-terminaux, on va appeler récursivement une procédure qui consiste à :

- Calculer l'arbre d'e.r. résultant du non-terminal au sommet de la pile
- Déterminer si l'autre non-terminal correspond à une opération valide, et le cas échéant, déterminer l'arbre qui en résulte (une opération à la racine et un de ses fils correspondant au 2ème opérande de l'opération).

On fusionne ensuite ces deux arbres, le 1er arbre devenant fils de la racine du 2ème arbre. (on renvoie juste le 1er arbre si l'autre non-terminal ne correspond pas à une opération)

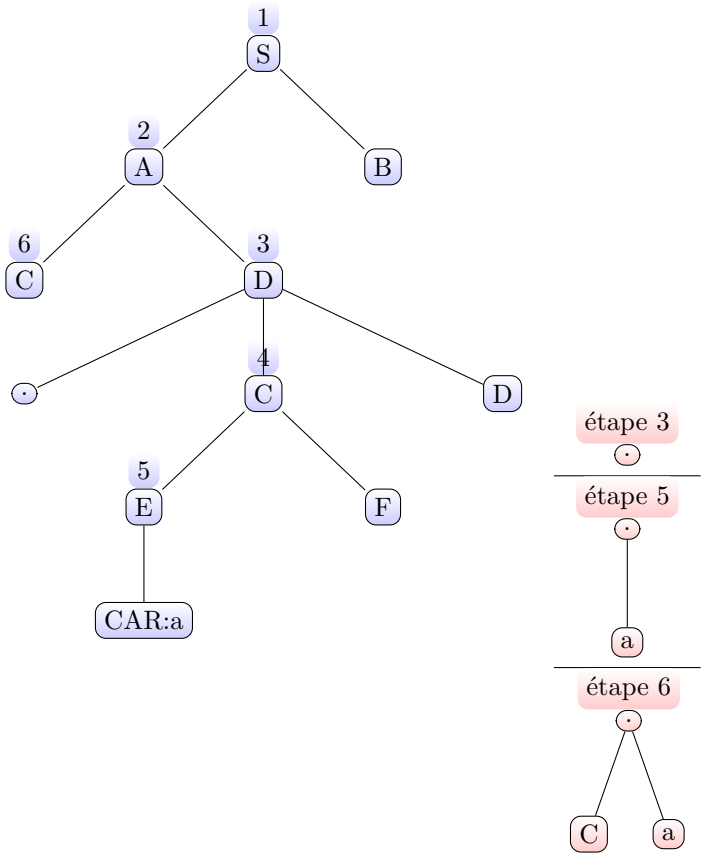

Figure 2: Exemple de construction d'un arbre d'e.r. à partir d'un arbre de dérivation (ordre des appels récursifs donné sur l'arbre de dérivation en bleu)