Projet Théorie Des Langages

Automate déterministe :

Lecture d'un mot par un automate :

- l'automate étant déterministe la lecture et la transition entre les états se fait simplement
- la lecture est bloquante, c'est à dire lorsque l'automate entre dans un état acceptant la lecture se termine et annonce que le mot w a été trouvé sinon il continue de lire

Création d'un automate qui reconnaît un mot w dans une chaîne :

- la fonction **create_automate** permet 'd'instancier' l'AFD, à ce niveau là on connaît déjà des informations comme l'alphabet (**sigma**), le nombre d'état (**nQ**), qui correspond à la longueur du mot w + 1, et l'état initial (**init**) , pour l'application correspondante à la liste des état (**e**) elle est appelé par une fonction auxiliaire **e_automate**.

- la fonction **e_automate** permet de donner un **etat** à un **int** (int → etat)
  lors de son premier appel dans la fonction **create_automate** elle se lance avec la variable **e_empty**, qui est une version vide de l'application e, la fonction **e_automate** va venir compléter cette application en faisant correspondre le numéro de l'état (**int**) avec un élément du type **etat**. Pour ce faire il s'appelle récursivement en fonction du nombre d'états nQ. Un état contient une fonction de transition qui défini en fonction de l'état actuel et du caractère passé en paramètre le numéro de l'état suivant. Cette application de transition (**t**) est complété par la fonction **t_automate.**

- la fonction **t_automate** permet de donner un état (**int**) à un caractère (**char**) en fonction de l'état actuel (**etat**). lors de son premier appel dans la fonction **e_automate** elle se lance avec la variable **t_empty**, qui est une version vide de l'application e. Ensuite elle parcours l'alphabet et complète l'application avec des filtres sur chaque lettres de l'alphabet, par conséquent l'automate créé sera évidemment complet. **t_automate** utilise une fonction **chooseEtat.**

- la fonction **chooseEtat** permet en fonction de l'état actuel, du caractère passé en paramètre et de sa place dans le mot **w**, de calculer l'état suivant. Cette fonction se base sur une étude d'un automate créé à la main avec un mot w : 'abb' qui permet de constater des régularité ce qui a permis de créer la fonction.

Détail sur la fonction de changement d'état :

-On remarque que si l'état actuel est 1 et que le caractère passé en paramètre est la première lettre de w on passe alors dans l'état i + 1, ainsi de suite si l'état actuel est l'état actuel est 2 et que le caractère passée en paramètre est la seconde lettre de w alors on passe dans l'état 3. Et ainsi de suit pour les autres lettre du mot w, ce que qui nous permet d'obtenir un nombre d'état (**nQ)** égal à la longueur du mot w + 1. Admettons maintenant l'automate construit pour permettre de reconnaître les chaînes contenant le mot 'abb'. Si l'on apllique cette lecture sur la chaînes 'abab' alors le parcours des état sera le suivant :

1 → 2 → 3 → 2 → 3

Lors de la lecture du second ’a’ l’automate revient sur l’état 2 car la lettre a est la première lettre du mot w par conséquent.

En somme on obtient la fonction suivante :

```
let chooseEtat (car : char) (w : string) (etat : int) = match car with
        c when (c = w.[0]) -> if c != w.[etat-1]  (1)
                then 2
                else etat + 1
        |c -> if c = w.[etat-1] (2)
                then etat + 1
                else 1;;
```

(1) : filtre si c est égal au premier caractère de w et différent du caractère de w à l position état actuel -1 alors on revient sur l état 2, sinon c’est que l’on poursuit la reconnaissance d’une éventuelle présence de w dans la chaîne.

(2) : dans tout les autre cas si le caractère est attendu par l’automate il passe en i + 1 sinon il retombe dans l’état 1.

Cette technique permet sans faillir de couvrir l’ensemble des mot w, il n’est pas sensible au répétition.

Test :

- la partie test dans le fichier AFD.ml teste le création de l’automate à partir de différents mot w. Ces automates sont testé sur un chaîne basé sur l’alphabet {‘A’, ‘C’, ‘G’, ‘T’} contenant 250 000 lettres. La dernière ligne mise en commentaire permet de lancer la lecture des differents automate sur cette chaînes, elle dure environ 5 minutes et renvoie pour chacun des automates, la présence du mot w et le temps de lecture. Pour un mot w qui se trouve à la fin de la chaîne des 250 000 caractères la lecture prend environ 50 secondes.