TP2 : Opérations régulières

1 Objectif

L'objectif de ce TP est de programmer les opérations d'union et de concaténation de deux automates, ainsi que la fermeture de Kleene d'un automate. On programmera aussi une fonction permettant de construire un automate élémentaire reconnaissant un seul symbole.

Pour cela, nous utiliserons la librairie `Automaton.c`.

Afin de tester les fonctions à programmer, on programmera un interpréteur, dont le squelette vous est donné dans le fichier `interpreteur.c`

2 Les fonctions à programmer

- `Automaton *elem(int symbol);` constuit un automate qui reconnaît le langage composé d'un seul mot lui même composé du symbole donné en argument à la fonction.
- `Automaton *unio(Automaton *a1, Automaton *a2)` constuit un automate qui reconnaît l'union des langages reconnus par `a1` et `a2`. Cette fonction n'est pas destructrice : à l'issue de l'appel `union(a,b)`, les automates `a` et `b` sont inchangés. Ceci est aussi vrai des deux fonctions suivantes.
- `Automaton *concat(Automaton *a1, Automaton *a2)` constuit un automate qui reconnaît la concaténation des langages reconnus par `a1` et `a2`.
- `Automaton *kleene(Automaton *a);` constuit un automate qui reconnaît la fermeture de Kleene du langage reconnu par `a`.

Ces fonctions devront **impérativement** se trouver dans le fichier `reg_op.c`. Vous modifierez le `Makefile` en conséquence.

3 L'interpréteur

L'interpréteur permet de construire des automates, de réaliser des opérations sur ces derniers, et de les sauvegarder.

Il utilise une pile ainsi que des registres. La pile permet de stocker les automates sur lesquels sont effectués les opérations. Ainsi, lorsqu'on effectue une opération

d'union de deux automates, les arguments de cette opération sont les deux automates qui se trouvent au sommet de la pile. Ils sont alors dépilés et le résultat de l'union est emplié.

Les registres permettent de stocker les automates construits dans la pile afin de pouvoir s'en servir plus tard.

L'interpréteur est constitué d'une boucle infinie (`while(1)`) qui lit sur le flux d'entrée des commandes et appelle les fonctions qui permettent de réaliser les commandes reconnues.

Voici la liste des commandes déjà programmées :

- `stack` : affiche le nombre d'éléments présents dans la pile
- `print` : affiche l'automate en sommet de pile
- `draw` : dessine l'automate en sommet de pile
- `save <str>` : sauvegarde l'automate en sommet de pile dans le fichier `<str>`
- `store <int>` : copie l'automate en sommet de pile à l'adresse `<int>`
- `load <int>` : empile l'automate se trouvant à l'adresse `<int>`
- `help` : affiche la liste des commandes
- `quit` : arrête l'execution de l'interpréteur

et voila les commandes que vous devez programmer :

- `elem <int>` : crée un automate reconnaissant `<int>`
- `union` : effectue l'union des deux automates en sommet de pile
- `concat` : effectue la concaténation des deux automates en sommet de pile
- `kleene` : effectue la fermeture de kleene de l'automate en sommet de pile