Rapport de Projet : Égalité d'expressions régulières

Binôme : BIGNOLLES – ACHAIBOU

1. Présentation du projet

Le but de ce projet est de fabriquer un programme capable de dire si deux expressions régulières sont équivalentes ou non. Pour y arriver, nous avons séparé le travail en deux parties distinctes :

- Une partie écrite en langage C qui s'occupe de lire et comprendre ce que l'utilisateur écrit.
- Une partie écrite en Python qui réalise tous les calculs sur les automates.

L'alphabet utilisé dans le projet contient les lettres a, b et c. La lettre E est utilisée pour représenter le mot vide (epsilon).

2. Comment le programme lit les expressions (Langage C)

Pour analyser les expressions régulières écrites par l'utilisateur, nous avons utilisé deux outils classiques : Lex et Yacc.

Le fichier regexp.l :

Ce fichier sert à reconnaître les différents caractères de l'expression. Lorsqu'il détecte une lettre ou un symbole comme * ou +, il transmet cette information à la suite du programme.

Le fichier regexp.y :

Ce fichier définit les règles de construction des expressions régulières. Par exemple, il précise que la concaténation (le point .) est prioritaire sur l'addition (+).

La traduction vers le Python :

À chaque fois que le programme C comprend une partie de l'expression, il écrit une ligne de code Python dans un fichier appelé main.py. Ce fichier permet de faire le lien avec la partie Python du projet.

3. Création et traitement des automates (Python)

Dans le fichier automate.py, nous avons écrit plusieurs fonctions permettant de transformer les expressions régulières en automates de Thompson.

Les fonctions de base :

Nous avons implémenté des fonctions pour l'union, la concaténation et l'étoile de Kleene. Ces opérations permettent de construire progressivement les automates.

La simplification des automates :

Afin de pouvoir comparer deux automates, ils doivent être simplifiés. Pour cela, nous utilisons la fonction tout_faire qui supprime les transitions epsilon, rend l'automate déterministe, le complète et enfin le minimise.

4. Test d'égalité et résultats

Une fois les automates simplifiés, nous pouvons tester l'égalité des deux expressions. Si les automates obtenus sont identiques, alors les expressions sont équivalentes.

Nous avons testé les expressions (a+b)*.c et a*.c + b*.c. Le programme a généré automatiquement le code Python, construit les automates et affiché le résultat NON EGAL.

Ce résultat est correct, car la première expression autorise un mélange de lettres 'a' et 'b' avant le symbole 'c', alors que la seconde oblige à choisir uniquement des 'a' ou uniquement des 'b'.

5. Conclusion et utilisation du programme

Pour simplifier l'utilisation du projet, nous avons créé un Makefile. Grâce à lui, il n'est pas nécessaire de taper de nombreuses commandes manuellement. Une simple commande make test permet de compiler le programme en C, de générer le code Python et d'obtenir le résultat final. Cela nous a permis de vérifier facilement le bon fonctionnement du projet.