Éditeur Automates (AEF)

Le projet du Groupe 1 composé de SILVA SAMUEL, ZIDI YANIS, NAHIL EL BEZZARI, ENZO EDMOND pour l'année ing1GIA 2023-2024.

Description

AEF = {'etat1':{'transition':[{'name':'' '', 'dest':' '}, {'name': '', 'dest':' '} , ...], 'initial': bool, 'final': bool}, 'state2': ...}

Ce projet consiste en la création d'un éditeur textuel dédié aux Automates d'États Finis (AEF). Il permet de manipuler les AEF de diverses manières, y compris la création, l'importation, la modification, la sauvegarde et la suppression. De plus, l'éditeur offre des fonctionnalités avancées pour interagir avec les AEF, telles que la vérification de reconnaissance de mots, miroir, modifier un aef, concaténation...

Fonctionnalités

1. **Créer AEF**: Permet de créer un nouvel automate à états finis.
2. **Modifier AEF**: Ouvre un automate existant pour le modifier.

Renommer AEF: Change le nom d'un automate sélectionné.

3. **Supprimer AEF**: Efface un automate de la liste ou de la mémoire.
4. **Complément**: Crée le complément d'un automate, inversant ses états finaux.
5. **Miroir**: Inverse le sens des transitions pour créer un automate miroir.
6. **Produit**: Combine deux automates pour créer un produit des deux.
7. **Concaténation**: Joint deux automates de manière séquentielle.
8. **Reconnaissance de mot**: Vérifie si un mot est reconnu par l'automate.
9. **Étoile**: Applique l'opération étoile de Kleene sur l'automate pour reconnaître les répétitions.
10. **Importer un AEF**: Charge un automate depuis un fichier externe.
11. **Exporter un AEF**: Sauvegarde l'automate actuel dans un fichier.

Installation

Prérequis

Assurez-vous d'avoir Python installé sur votre système. Ce projet est développé avec Python 3.8, mais il devrait être compatible avec d'autres versions de Python 3.

Étapes d'Installation

1. **Cloner le Répertoire GitHub** : Clonez d'abord le dépôt GitHub où se trouve le projet.
2. **Installer Graphviz** : Graphviz est nécessaire pour la génération de graphes d'états.
3. **Installer CustomTkinter** : CustomTkinter est utilisé pour l'interface utilisateur. Pour l'interface on a utilisé CustomTkinter une version modifié open-source de Tkinter trouvable sur Github. Elle permet de faire des interfaces plus fluides et plus belles.
4. **Vérifier les Dépendances** : Assurez-vous que toutes les dépendances sont correctement installées en exécutant les commandes suivantes dans votre console Python.
5. **Lancement de l'Application** : Après avoir installé toutes les dépendances, lancez le fichier `main.py` pour démarrer l'application.

Résolution des Problèmes

Si vous rencontrez des problèmes lors de l'installation, vérifiez que vous avez la bonne version de Python et que toutes les dépendances sont installées. Consultez également la documentation des bibliothèques individuelles pour des instructions spécifiques à votre système d'exploitation.

Utilisation

Programme est lancée en exécutant `main.py`. Elle fonctionne avec une interface en mode console, fournissant des instructions interactives pour manipuler les AEF.