Rapport du projet

Yuntian Shen & Siman Chen

Objectif du Projet

Ce programme, intitulé “TM Turning Machine tokenizer and parser”, est conçu pour analyser, interpréter et représenter visuellement des instructions pour une machine de Turing décrites dans le format spécifique des règles “MTdV”. Les résultats sont exportés sous forme de fichiers JSON et de graphes visuels.

Résumé Fonctionnel

Le programme effectue les étapes suivantes :

1. **Lecture et Tokenisation** :
   - Lit un fichier d’entrée contenant les instructions.
   - Analyse et segmente le texte en une liste de **tokens** en supprimant les commentaires et en corrigeant certains cas particuliers (comme } attaché à d’autres mots).
   - Les tokens sont des éléments lexicaux individuels comme I, P, G, D, ou des blocs structurés comme si (0).
2. **Analyse Syntaxique** :
   - Les tokens sont transformés en une liste de **nœuds** (instructions de la machine de Turing).
   - Ces nœuds incluent leurs relations (transitions entre instructions), permettant de représenter un flux d’exécution.
3. **Génération des Sorties** :
   - Exporte les nœuds et leurs relations dans un fichier JSON pour un traitement ou une analyse ultérieure.
   - Génère un graphe visuel (fichier image .png) des instructions et des transitions entre elles.

Structure et Fonctionnalités Principales

1. Classes et Méthodes

- Node :

- Une classe définissant les nœuds de la machine.
- Attributs : id (identifiant), inst (instruction associée), edges (transitions vers d'autres nœuds).

- MTdV :
  - La classe principale, encapsulant les processus de tokenisation, analyse et génération des sorties.

Méthodes importantes :

- tokenize :
  - Transforme le texte brut en une liste de tokens, gérant les cas particuliers et supprimant les commentaires.
- parse :
  - Convertit les tokens en une structure de nœuds et de relations (graphe de la machine).
  - Implémente une logique basée sur des piles (stack, stack_b) pour gérer les blocs conditionnels (si) et les boucles (boucle).
- generate_json :
  - Exporte les instructions et leurs relations au format JSON.
- generate_graph :
  - Crée une représentation visuelle des instructions à l'aide de PyGraphviz.

2. Gestion des Instructions

Chaque instruction du format MTdV est interprétée selon sa signification :

- I, P, G, D, 0, 1 : Actions simples (afficher, écrire, déplacer, etc.).
- si (0) / si (1) : Structures conditionnelles.
- boucle, fin : Boucles.

- # : Marqueur de fin de fichier.

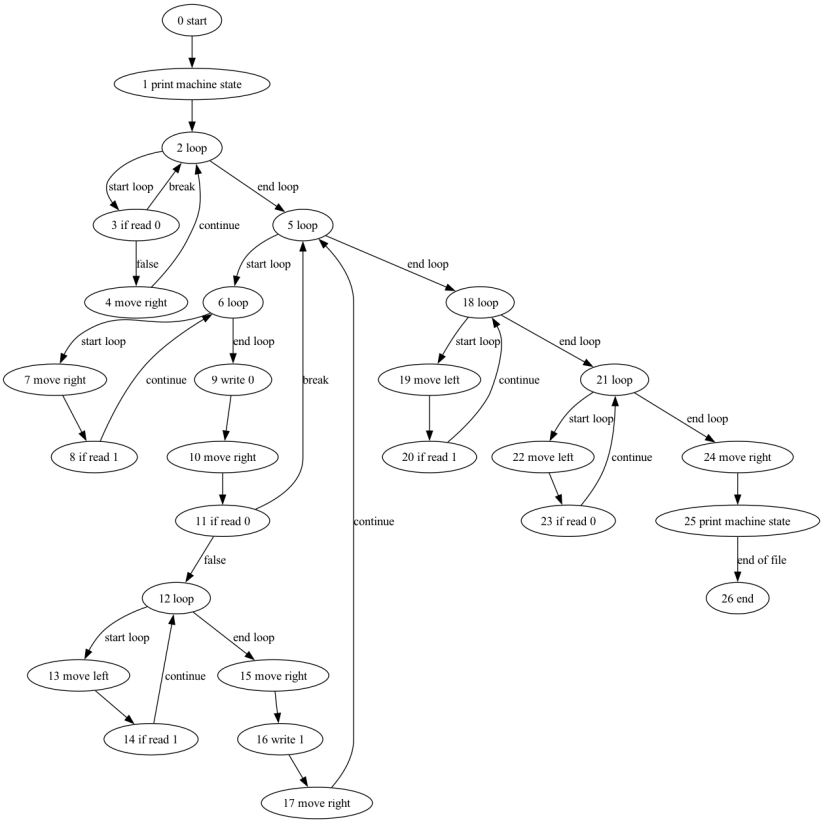
Les erreurs comme des accolades mal fermées (}) ou un fichier sans marqueur de fin sont gérées avec des exceptions.

3. Arguments CLI

- input : Chemin du fichier d'entrée contenant les instructions.
- -o/--output : Dossier ou chemin pour sauvegarder les fichiers générés.

4. Sorties

- **JSON** :
    - Structure les nœuds et leurs relations dans un fichier au format JSON, utile pour des analyses automatisées.
- **PNG** :
    - Génère un graphe orienté représentant visuellement les transitions d'instructions.



Un example pour addition.1.TS

Résumé
Le programme est un analyseur robuste et extensible pour un langage spécifique, offrant des sorties adaptées pour l’analyse et la visualisation. Avec des ajustements mineurs, il pourrait être généralisé pour d’autres formats ou applications.