https://github.com/cs-ubbcluj-ro/lab-work-computer-science-2024-TeodoraVlad12/tree/main/2-Finite-Automata

FA Documentation

Class Structure

1. FA Class

- **Attributes:**
  - `states`: A list that holds all states in the finite automata
  - `alphabet`: A list containing the alphabet (input symbols) recognized by the FA.
  - `initialState`: A string representing the initial state of the FA.
  - `finalStates`: A list of final (accepting) states in the FA.
  - `transitions`: A list of tuples representing state transitions in the format `(current_state, next_state, symbol)`.
  - `filename`: A string that stores the file path from which the FA configuration is read.

Methods

1. `__init__(filename)`

- **Description**: Initializes a new instance of the `FA` class with an empty configuration and sets the filename.
- **Parameters**:
  - `filename`: The path to the file containing the FA configuration.

2. `ReadFile()`

- **Description**: Reads and loads the FA configuration from the specified file. This includes states, alphabet, initial state, final states, and transitions.
- **File Format**:
  - The file should contain the following lines:
    - `states`: A comma-separated list of states.
    - `alphabet`: A comma-separated list of symbols in the alphabet.
    - `initial state`: The initial state.
    - `final states`: A comma-separated list of final states.
    - `transitions`: Comma-separated transitions in the format `current_state next_state symbol`.
- **Functionality**:
  - Reads and splits each line based on a colon `:` and stores the values in the corresponding attributes.

3. **`PrintStates()`**

- **Description**: Prints all states in the FA.

4. **`PrintAlphabet()`**

- **Description**: Prints the alphabet symbols recognized by the FA.

5. **`PrintFinalStates()`**

- **Description**: Prints the final (accepting) states in the FA.

6. **`PrintInitialState()`**

- **Description**: Prints the initial state of the FA.

7. **`PrintTransitions()`**

- **Description**: Prints each transition in the FA in the format `current_state -> next_state : symbol`.

8. **`CheckAccepted(word)`**

- **Description**: Checks if a given input `word` is accepted by the FA by processing each symbol through the state transitions.
- **Parameters**:
    - `word`: A string representing the input sequence to check.
- **Returns**: `True` if the word is accepted (i.e., the final state reached is in `finalStates`), otherwise `False`.
- **Logic**:
    - Starts from `initialState`.
    - For each symbol in `word`, searches for a matching transition.
    - If no transition is found for the symbol, returns `False`.
    - If a valid final state is reached after processing the word, returns `True`.