Grammar Parser - LL(1) and SLR(1) Analyzer

Erick Eduardo Guerrero, Juan David Ortiz, Anyela Cristina Jimenez

Description

A Python-based tool developed for the *Formal Languages and Compilers* course (ST0270 / SI2002). This tool implements both **LL(1)** and **SLR(1)** parsers for context-free grammars. It features colorful terminal output, detailed parsing steps, and comprehensive grammar analysis.

Group Members

- Erick Eduardo Guerrero
- Juan David Ortiz
- Anyela Cristina Jimenez

Environment

- **Operating System:** Windows 11
- **Programming Language:** Python 3.12
- **Libraries Used:**
  - `tabulate` 0.8.9
  - `colorama` (optional, for color output)

Features

Grammar Analysis

- Automatic detection of grammar type: **LL(1)**, **SLR(1)**, both, or neither
- Proper computation of `FIRST` and `FOLLOW` sets
- Detection of left recursion
- Shift/reduce conflict detection (SLR(1))
- User-friendly grammar validation

Parsing Capabilities

LL(1) Parser:

- Parsing table generation
- Step-by-step derivation output
- Halts and warns on left recursion

SLR(1) Parser:

- LR(0) automaton construction
- `ACTION` and `GOTO` table generation
- Shift/reduce conflict detection

User Interface

- Colored terminal output (`colorama`)
- Formatted tables (`tabulate`)
- Interactive parser selection and clear diagnostics

Usage

Step 1: Prepare `grammar.txt` file

```
<number of productions>
Production 1
Production 2
...
Production N
<string to analyze 1>
<string to analyze 2>
...
<string to analyze M>
```

Example:

```
3
S -> AB
A -> aA | d
B -> bBc | e
adbc
d
a
```

Step 2: Run the Analyzer

```
python Main.py grammar.txt
```

Step 3: Select Parser Type

- `T`: LL(1) parser
- `B`: SLR(1) parser
- `Q`: Quit

File Structure

```
project/
          Main.py              # CLI and main entry point
          F.py                 # LL(1) parser module
          S.py                 # SLR(1) parser module
          grammar_utils.py     # Grammar and set processing
          table_utils.py       # Table rendering tools
          grammar.txt          # Input grammar file
          README.md            # Documentation
```

Sample Output

Grammar Analysis:

```
    Grammar is LL(1) - No conflicts found
    Grammar is SLR(1) - No conflicts found



 Grammar is both LL(1) and SLR(1)
```

Parsing Result:

```
      Result: YES
    Input Accepted
```

Problems Encountered and Solutions

1. Left Recursion

Problem: Infinite loop in LL(1) parsing.
Solution: Added detection and halting mechanism.

2. Incorrect FIRST and FOLLOW

Problem: Improper propagation of epsilon (ε).
Solution: Fixed computation logic.

3. Shift/Reduce Conflicts

Problem: Conflicts in SLR(1) table.
Solution: Conflicts are flagged, parsing is stopped.

4. Table Construction Errors

Problem: Uninitialized dictionaries.
Solution: Improved setup and added debugging tools.

5. Invalid Grammars

Problem: Ambiguous or unfit grammars.
Solution: Grammar is classified, parsing prevented.

Requirements

- Python 3.8+
- `tabulate`
- `colorama` (optional)

Limitations

- Only supports LL(1) and SLR(1)
- Grammar format must be strictly followed
- No support for ambiguous grammars or precedence resolution