LL(1) and SLR(1) Grammar Parser: Problems And Solutions

Samuel Calderon Duque
Matias Monsalve Ruiz

Introduction

During the development of the grammar parser supporting LL(1) and SLR(1) analysis, several issues were encountered related to grammar input handling, parsing logic, and table construction. This document summarizes the most important problems and how they were solved.

1. Ambiguous Input Format

Problem

Initially, the program expected each production in the form `A -> b | c`, but did not handle multiple right-hand sides split by `|`. This caused all alternatives to be interpreted as a single string.

Solution

Each production must be separated with a blank space. We updated the grammar input instructions and parsing logic to expect:

```
A -> b c
```

2. Incorrect FIRST and FOLLOW Set Computation

Problem

Some FIRST and FOLLOW sets were incomplete or incorrect due to missing epsilon (ϵ) propagation, especially for nullable productions.

Solution

We implemented an iterative algorithm for FIRST and FOLLOW computation that:

- Adds ϵ to FIRST if a production can derive the empty string.
- Correctly propagates FOLLOW sets through chains of nonterminals.

3. LL(1) Table Conflicts

Problem

The LL(1) table construction sometimes failed because multiple productions matched the same table entry, indicating a grammar conflict.

Solution

We added logic to detect such conflicts early. If a duplicate key was detected in the LL(1) table, the grammar was marked as non-LL(1), and parsing fallback to SLR(1) was allowed when possible.

4. Incorrect LR(0) State Transitions

Problem

The SLR(1) parser occasionally produced wrong results due to incorrect state transitions or missing closures in the LR(0) automaton.

Solution

We carefully reviewed the `closure()` and `goto()` functions to:

- Include all valid items during closure.
- Ensure transitions correctly reflect item movements.
- Use `frozenset()` to uniquely identify and store states.

5. Shift/Reduce Conflicts in SLR(1)

Problem

Some grammars caused shift/reduce conflicts in the SLR(1) ACTION table.

Solution

We:

- Verified that FOLLOW sets were correctly used when inserting reduce actions.
- Ensured that no conflicting shift and reduce entries existed in the same table cell.
- Marked the grammar as non-SLR(1) if any conflict was detected.

6. Incorrect Parsing Behavior

Problem

Valid strings were sometimes rejected or incorrect strings were accepted.

Solution

Parsing logic was reviewed to ensure:

- Proper use of stack-based simulation in both LL(1) and SLR(1) modes.
- Accurate token matching and symbol popping.

Conclusion

These debugging steps helped improve the parser's functionality.