Documentación de Problemas y Soluciones del Proyecto LL(1) y SLR(1)

Emmanuel Alvarez Castrillon y Samuel Herrera Galvis - Lenguajes Formales

Mayo 14, 2025

Introducción

Durante el desarrollo del proyecto se implementaron algoritmos para el cálculo de los conjuntos First y Follow, y para la construcción e interpretación de analizadores sintácticos LL(1) y SLR(1). Si bien gran parte del proyecto se completó sin mayores contratiempos, surgieron varios problemas durante la implementación y validación de los analizadores, especialmente en lo referente al correcto reconocimiento de cadenas, según el comportamiento esperado para cada tipo de gramática.

Problema Principal Encontrado

Uno de los principales retos se presentó con la ejecución del **Caso 1**, definido por el enunciado como:

> "La gramática es SLR(1) y LL(1). Imprimir: 'Seleccionar un analizador (T: para LL(1), B: para SLR(1), Q: salir):', luego recibir cadenas para analizar hasta que se proporcione una línea vacía."

Comportamiento Incorrecto Inicial

Al comenzar a probar con una gramática que cumplía con las condiciones de ser tanto LL(1) como SLR(1), observamos que la ejecución era defectuosa en el modo SLR(1). Más específicamente:

- Las cadenas ingresadas eran correctamente evaluadas por el analizador LL(1), devolviendo salidas "yes", "yes" y "no", según lo esperado.
- Sin embargo, al seleccionar el analizador SLR(1), todas las cadenas ingresadas eran incorrectamente rechazadas, mostrando "no", "no", "no", incluso para aquellas válidas según la gramática.

Diagnóstico y Solución

Tras una revisión del código, se detectaron las siguientes posibles causas del fallo:

1. El analizador SLR(1) no interpretaba correctamente el símbolo de fin de cadena `$`, lo que provocaba un rechazo prematuro en muchos casos.
2. Había problemas en el manejo de las acciones "shift", "reduce" y "accept" dentro de la tabla ACTION construida a partir de la colección canónica de elementos LR(0).

3. La estructura de la pila y el procesamiento de símbolos en la función `parse` del archivo `slr1_parser.py` no seguía correctamente la lógica del algoritmo SLR(1).

Para resolverlo, se tomaron las siguientes medidas:

- Se revisó y corrigió el algoritmo de construcción de las tablas ACTION y GOTO, asegurando que se generaran correctamente las transiciones y acciones de reducción.
- Se implementó una verificación más estricta de los símbolos en las producciones, prestando especial atención a cómo se representaban epsilon (`e`) y el símbolo de fin de entrada (`$`).
- Se depuró detalladamente la función de análisis, utilizando trazas para cada transición de la pila y la lectura de tokens, permitiendo aislar errores de lógica.

Luego de múltiples iteraciones y ajustes, el comportamiento del analizador SLR(1) fue corregido y pasó a reflejar con precisión las decisiones esperadas: aceptando o rechazando las cadenas de prueba de acuerdo a la gramática, igual que el analizador LL(1).

Otros Problemas Menores

- **Separación incorrecta de producciones:** En etapas tempranas, el análisis del input fallaba con producciones con múltiples alternativas debido a una segmentación incorrecta. Esto se resolvió reformateando el método de lectura y normalización de la gramática para dividir correctamente alternativas separadas por espacios y barras verticales.
- **Errores con producciones vacías:** Inicialmente, las producciones con epsilon no se trataban adecuadamente, lo que afectaba los conjuntos First y Follow. Se mejoró el manejo de estas producciones usando tuplas vacías y condiciones explícitas.
- **Formateo de salida:** Las respuestas del programa no coincidían con los ejemplos requeridos. Se estandarizaron las salidas para que fueran "yes" o "no", en minúscula, como pedía el ejemplo 2 del enunciado.

Conclusiones

La implementación del proyecto permitió profundizar en el entendimiento de los conceptos fundamentales de análisis sintáctico. Resolver los problemas asociados al funcionamiento del analizador SLR(1) fue particularmente enriquecedor, ya que implicó una revisión minuciosa de los conceptos de autómatas LR, el manejo de estados, y el uso correcto de Follow en acciones de reducción. El sistema final ahora permite analizar cualquier gramática adecuada y clasificarla como LL(1), SLR(1), ambas o ninguna, ejecutando correctamente los analizadores según el caso.