Proyecto TLP02-2025. Sección 02.

Gramáticas formales y algoritmos modernos: enlace entre los lenguajes formales y el procesamiento de lenguaje natural

Introducción

Este proyecto busca que los estudiantes comprendan las bases teóricas de los lenguajes formales y sus limitaciones frente al lenguaje natural, así como las diferencias fundamentales entre los parsers tradicionales y las técnicas modernas de Procesamiento de Lenguaje Natural (NLP). Esto se desarrollará en dos fases complementarias, de manera grupal (3 a 5 integrantes), combinando investigación teórica, implementación práctica y comparación crítica con herramientas actuales de IA.

El parser o el analizador sintáctico se encarga de revisar el texto de entrada con base a una sintaxis definida, utilizando métodos sistemáticos basados en gramáticas libres del contexto. Por otro lado, está el analizador semántico que es la pieza de software de un compilador que se encarga de detectar errores semánticos, como que los tipos que intervienen en las expresiones sean compatibles o que los parámetros reales de una función sean coherentes con los parámetros formales, es decir, en el análisis semántico se dota de un significado coherente a lo que se ha hecho en el análisis sintáctico.

La idea de este proyecto es mantener el corazón de compiladores y lenguajes de programación basados en gramáticas libres del contexto, pero abrir la ventana al mundo de NLP y la IA, y que se vea su aplicación en sistemas modernos.

Fase 1 – Lenguajes formales vs. lenguaje natural (40%)

Objetivo:

- Entender por qué los lenguajes de programación suelen modelarse con gramáticas libres de contexto (CFG), mientras que los lenguajes naturales exceden ese marco.
- Experimentar con parsers clásicos y comprobar sus limitaciones frente a frases en español o inglés.
- Entender la diferencia de expresividad y complejidad.

Resumen de Fase: escribir un pequeño informe + demo (ejemplo en Python con NLTK o spaCy) mostrando cómo un parser LL(1) puede analizar una gramática formal, pero falla en lenguaje natural.

Actividades:

1. Investigar y explicar las diferencias clave entre lenguajes formales y naturales (estructura, ambigüedad, recursividad, contexto).

2. Implementar un parser sencillo (LL(1) o descendente recursivo) en Python que analice una pequeña gramática formal para lenguaje C estándar o C++
3. Probar el mismo parser con frases en lenguaje natural y documentar las fallas.
4. Realizar una demo apoyada en una librería de NLP (NLTK o spaCy) para mostrar la diferencia en la capacidad de análisis.

Entregables:

- Informe corto (5–7 páginas) con el análisis comparativo.
- Código fuente del parser formal + ejemplos de prueba.
- Presentación grupal (15–20 min) explicando hallazgos y demostración.

Fecha de entrega: 6 de noviembre.

Fecha de presentaciones: 11 y 13 de noviembre.

Fase 2 – Mini-parser para lenguaje natural limitado (60%)

Objetivo:

- Diseñar e implementar un parser propio para un subconjunto simplificado del español o inglés.
- Contrastar el desempeño de dicho parser con el de un modelo moderno de NLP basado en estadística o deep learning.

Actividades:

1. Definir una gramática libre de contexto limitada, por ejemplo:
    - Oraciones en forma Sujeto–Verbo–Objeto con adjetivos o modificadores sencillos.
    - Restricción a un vocabulario reducido.
2. Implementar un parser descendente recursivo que valide frases según esa gramática.
3. Probar el parser con ejemplos válidos e inválidos.
4. Comparar resultados con una herramienta moderna (ej. spaCy, HuggingFace Transformers, o servicios cloud de NLP).
5. Analizar diferencias: robustez, ambigüedad, escalabilidad y aplicabilidad práctica.

Entregables:

- Informe final (7-9 páginas) documentando diseño de la gramática, implementación, pruebas y comparación con NLP moderno.
- Código fuente del parser + ejemplos de entrada/salida.
- Presentación final (15–20 min) con demo en vivo.

Fecha de entrega: 23 noviembre

Fecha de presentaciones: 24 y 25 de noviembre.