Relazione sull'utilizzo dell'AI nel Progetto di Ingegneria dei Linguaggi di Programmazione

Angelo Barone
Matricola: NF22500157
Corso: Ingegneria dei Linguaggi di Programmazione

Anno Accademico: 2025/2026

1 Introduzione

Il presente documento descrive le modalità, l'efficacia e le criticità riscontrate nell'utilizzo di strumenti di Intelligenza Artificiale Generativa (LLM) durante lo sviluppo del compilatore MiniLang. L'AI è stata integrata nel flusso di lavoro non come sostituto della logica progettuale, ma come *Thought Partner* (partner di pensiero) e acceleratore per attività ripetitive.

2 Tipologie di Utilizzo

L'assistente AI è stato impiegato trasversalmente nelle diverse fasi del ciclo di vita del software:

2.1 Supporto nell'implementazione del Codice

L'AI ha supportato la scrittura di boilerplate code e l'implementazione di nuove librerie e Pattern.

- **Visitor Pattern:** Supporto nella strutturazione delle classi `NodeVisitor`, per l'implementazione del Visitor Pattern con riflessione.
- **Backend LLVM:** Aiuto nella comprensione e nell'utilizzo della libreria `llvmlite`

2.2 Generazione e Validazione dei Test

L'AI è stata impiegata come strumento di supporto per garantire la robustezza del software tramite la generazione automatizzata di suite di test. L'utilizzo si è concentrato su tre aree metodologiche:

- **Automazione del Boilerplate:** Generazione rapida della struttura standard dei test unitari (setup, teardown, asserzioni) per i vari moduli del compilatore, riducendo il tempo dedicato alla scrittura di codice ripetitivo e permettendo di focalizzarsi sulla logica di validazione.

- **Identificazione di Edge Cases:** Supporto nell'ideazione di scenari limite e casi d'angolo (boundary cases) spesso trascurati nello sviluppo manuale, come input sintatticamente malformati, ricorsioni profonde o operazioni aritmetiche non valide, migliorando la resilienza del parser e dell'ottimizzatore.
- **Test di Integrazione:** Creazione di programmi sorgente completi nel linguaggio target (file `.mini`) progettati per sollecitare contemporaneamente più componenti dell'architettura (es. interazione tra scope delle variabili, chiamate a funzione e gestione della memoria), verificando la correttezza del flusso end-to-end.

2.3 Supporto all'Apprendimento

L'assistente virtuale ha svolto il ruolo di *Tutor Interattivo*, facilitando il collegamento tra i concetti teorici studiati nel corso e la loro implementazione pratica.

- **Comprensione di Librerie e Framework:** L'AI ha accelerato l'apprendimento relativo a librerie complesse (come i binding Python per LLVM), fornendo spiegazioni sul funzionamento delle API e suggerendo le best practices per la generazione del codice intermedio.
- **Scelte Architetturali:** Supporto nell'analisi dei trade-off progettuali (es. gestione della memoria sullo stack vs registri SSA, strategie di parsing), facendo da guida verso soluzioni implementative più adatte agli scopi progettuali.
- **Debugging Concettuale:** Oltre alla correzione degli errori di sintassi, l'AI è stata utilizzata per analizzare errori logici, fornendo spiegazioni dettagliate sulle cause aiutando a comprendere i principi di funzionamento dei compilatori.

3 Valutazione dell'Efficacia

3.1 Vantaggi

1. **Velocità di Sviluppo:** Riduzione drastica del tempo speso a scrivere codice ripetitivo (boilerplate) o a cercare la sintassi corretta di librerie complesse come `llvmlite`.
2. **Debugging Immediato:** Capacità di analizzare errori di compilazione e suggerire fix immediati.
3. **Qualità della Documentazione:** Supporto nella produzione di documentazione tecnica chiara, ben strutturata e grammaticalmente corretta.

3.2 Inconvenienti e Limiti

1. **Allucinazioni su Librerie:** In alcuni casi, l'AI ha suggerito metodi di librerie deprecati o inesistenti, richiedendo una verifica manuale sulla documentazione ufficiale.
2. **Necessità di Supervisione:** L'AI tende a generare codice "che funziona" ma non sempre ottimizzato o a volte non integrato correttamente con il resto del codice. È stato necessario revisionare l'architettura per garantire la correttezza logica del compilatore.

4 Conclusioni

L'integrazione dell'AI nel flusso di lavoro si è rivelata estremamente efficace. Ha permesso di concentrarsi sugli aspetti logici e architetturali di alto livello (come la progettazione dell'AST e la logica di ottimizzazione), delegando all'AI i dettagli implementativi di basso livello. L'uso dello strumento come "tutor" per chiarire concetti complessi di teoria dei compilatori ha rappresentato un valore aggiunto significativo per l'apprendimento.