C4 Compiler Analysis Report

Zayed AlDhaheri 100061222

Introduction

The C4 compiler is a C-based mini compiler, which translates source code into instructions which are then executed by the custom virtual machine (VM). This report will explain the core algorithms and concepts used in the lexical analysis, parsing, virtual machine execution, and memory management of the C4 compiler.

Lexical Analysis (Tokenization)

The lexical analyzer in C4 converts the source codes characters into tokens, which are small units of characters that contain keywords, operators, identifiers etc.

The next() function reads the characters from the source code (p). It skips whitespaces, newlines, and comments. It identifies keywords and assigns them a token type. For example, numbers (0..9), operators (+ - * / ** || == !=), Identifiers (a-z, A-Z, _) etc.

Parsing

C4 doesn’t use a full AST (Abstract Syntax Tree) like the usual C compilers. Instead, generation of the code and parsing happens at the same time.

The expr() function parses expressions like a + b * 2 by following operator precedence. The stmt() function parses statements like if, while, return, {…} blocks. Parsing rules are implemented using recursive descent parsing.

VM Execution

The c4 compiler has a custom VM that executes the instructions generated by the parser.

The VM fetches, decodes, and executes the instructions using a loop. These instructions operate on registers (pc, sp, bp, a) which stores, program counter, stack pointer, base pointer, and accumulator. Also it operates on stack which is used for function calls and local variables, and the data sections which stores global variables.

Memory Management in C

C4 manages memory using stack allocation for local variables and heap allocation for dynamic memory.

The stack memory (sp, bp) is used for function calls, local variables and parameters. The stack grows and shrinks with function calls ENT and LEV.

The heap memory (malloc, free) is used for dynamically allocating memory using MALC and freeing memory using FREE. The data section stores global variables.

Conclusion

The C4 compiler is a C-like compiler which implements lexical analysis, parsing, code generation, and execution in a compact way. It produces bytecode instead of assembly or machine code and it’s executed by its own virtual machine. Memory is manages using stack and heap allocation making the C4 compiler simple but function.