**Optimization Techniques:**

1. **Symbol Table Interaction:**
   * **Purpose:** The symbol table is used during code generation to manage variables.
   * **Functionality:**
     + When a variable is declared, the code inserts a new variable entry in the symbol table.
     + When a variable is used (in assignment or expression), the code looks up the variable from the symbol table.
   * **Semantic Analysis Implication:** The symbol table prevents using undeclared variables, because if a variable is not declared, then it won't be added to the symbol table, and it will not be able to find it during code generation.
   * **Limitation:** While the symbol table is *used*, the compiler *does not perform any semantic analysis* like checking data types, variable scope and other more advanced checks.
2. **Very Basic Error handling:**
   * **Purpose:**  
     The compiler will exit if the lexer or parser encounters an error, like invalid character, invalid statement and other errors.
   * **Functionality:**
     + The lex function will return empty vector if an invalid character is found in lexing stage.
     + The parse function will exit if parser fails.
   * **Semantic analysis implication**: These basic checks prevent compiler from compiling and generating assembly if a lexing/parsing error exists.
   * **Limitations:** The compiler does not check type mismatch errors and other semantically incorrect code.

**Why it's not a complete semantic analysis:**

* **No Explicit Type Checking:** The compiler does not do any checking to make sure variable types are consistent in the code, so variables can be declared, assigned and operated with any other values.
* **Limited Scope Handling:** The scope is always kept as zero, and local scoping is not implemented.
* **No Advanced Checks:** Features like function call checks, array checks, etc. are not implemented as the language and compiler itself is very basic.

**How to Add Semantic Analysis:**

To add true semantic analysis, the following needs to be done:

1. **Create a Semantic Analysis Phase:** Develop a new class or module dedicated to semantic analysis.
2. **Traverse AST:** The semantic analysis phase would take the Abstract Syntax Tree (AST) as input and traverse the tree.
3. **Type Checking:** Check for consistent type usage when variables are declared, assigned, or used in operations. The type of variables need to be stored in symbol table to perform checks.
4. **Scope Management:** Store the current scope of a variable, and perform lookups according to the scope. Add error messages if variables are used outside their scope.
5. **Error Reporting:** Generate informative error messages that are specific to semantic violations (e.g., "Type mismatch," "Undeclared variable in scope," etc.).