

Parse-time error identification and auto-correction from compiler error message using ML technique

Fasile Endalkachew
School of Electrical and Computer Engineering
Addis Abeba Institute of Technology
Addis Abeba Ethiopia

Abstract—Computer program source code is the list of human-readable instructions that a programmer writes often in a word processing program when he/she is developing some kind of program. And this source code is run through a compiler to turn into a machine code, also called object code. However, this compiler gives a parse-time (syntax) error message when the code does not conform to the syntax of the programming language this is also called compile time error for language such as C/C++ or Java. Ideally, these error message should be the easiest error to correct because the parser tells you exactly what is wrong and on what line the problem occurs, but in many cases it fails and the Programmers encounter cryptic compiler error messages that are difficult to understand and thus become hard to resolve. Because of one brace or an errant semicolon, the compiler may generate many errors throughout the program and this affect the performance of students, instructors or even the experienced programmers.

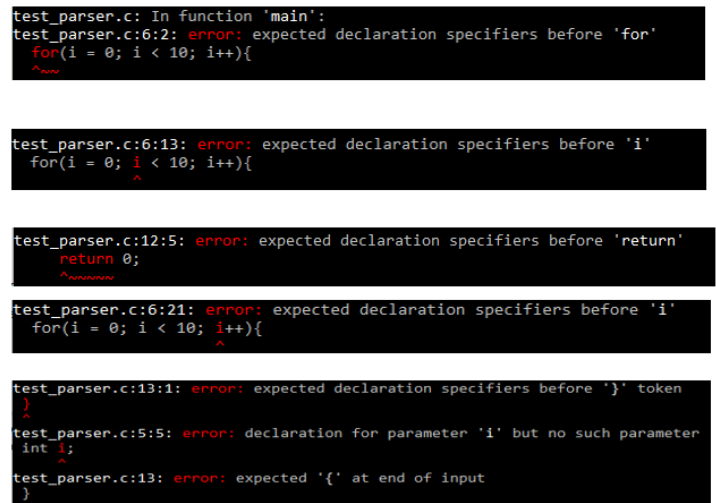
1. Introduction

Novice programmer often face cryptic compiler error messages due to different parse-time (syntax) errors and those errors are difficult to understand what it means and thus become harder to resolve. Many novices find it frustrating to struggle with those syntax errors [3] and waste their effort finding out the cause of error.

This paper propose that, it would be better to implement a model to identify and predict the exact cause of the errors by learning different compiler error messages using some machine learning approaches and finally try to fix the error.

1.1. Problem Statement

Computer program source code is the list of human-readable instructions that a programmer writes often in a word processing program when he/she is developing some kind of program. And this source code is run through a compiler to turn into a machine code, also called object code. However, this compiler gives a parse-time error message when the code does not conform to the syntax of the programming language this is also called compile time error for language such as C/C++ or Java.



```
test_parser.c: In function 'main':
test_parser.c:6:2: error: expected declaration specifiers before 'for'
  for(i = 0; i < 10; i++){
  ^~~~

test_parser.c:6:13: error: expected declaration specifiers before 'i'
  for(i = 0; i < 10; i++){
                ^

test_parser.c:12:5: error: expected declaration specifiers before 'return'
    return 0;
    ^~~~~~

test_parser.c:6:21: error: expected declaration specifiers before 'i'
  for(i = 0; i < 10; i++){
                    ^

test_parser.c:13:1: error: expected declaration specifiers before '}' token
}
^
test_parser.c:5:5: error: declaration for parameter 'i' but no such parameter
int i;
    ^
test_parser.c:13: error: expected '{' at end of input
}
```

Figure 1. Compiler error messages for one missing brace error

Normally, these error message should be the easiest error to correct because the parser tells you exactly what is wrong and on what line the problem occurs, but in many cases it fails and the Programmers encounter cryptic compiler error messages that are difficult to understand and thus become hard to resolve. Just because of one brace or an errant semicolon, the compiler may generate many errors throughout the program and this affect the performance of students, instructors or even the experienced programmers.

Figure 1 shows inconsistency of compiler error messages because of one single missing semicolon at different location. This inconsistency mainly happen because of poorly designed error message and this may negatively affect novice programmer more adversely.

1.2. Objective

1.2.1. General Objective. The general objective of this proposal is to identify and fix parse-time (syntax) error using machine learning approaches from compiler error messages

1.2.2. Specific Objective. predicting parse-time (syntax) errors from different compiler error messages using machine learning algorithm and statistical analysis.

1.3. Contribution

The contribution of this proposal is as follows. It implements a model that is used to identify and predict the exact cause of error using different compiler error message. After training the model and identifying the right cause of the error, then error will try to be fixed. And it also will help novice programmer to solve the frustration to struggle with those syntax error and also save their time on finding out the cause of errors. Which leads to enhance the productivity and quality of software.

2. Proposed Methodology

References

- [1] P. J. Brown, “Error messages: the neglected area of the man/machine interface,” *Communications of the ACM*, vol. 26, no. 4, pp. 246–249, 1983.
- [2] T. Flowers, C. A. Carver, and J. Jackson, “Empowering students and building confidence in novice programmers through gauntlet,” in *Proceedings of the 34th ASEE/IEEE Frontiers in Education Conference (FIE '04)*, vol. 1, pp. T3H-10–T3H-13, October 2004.
- [3] Kummerfeld, Sarah K., and Judy Kay. “The neglected battle fields of syntax errors.” In *Proceedings of the fifth Australasian conference on Computing education*-Volume 20, pp. 105-111. Australian Computer Society, Inc., 2003.
- [4] Jadud, Matthew C. “Methods and tools for exploring novice compilation behaviour.” In *Proceedings of the second international workshop on Computing education research*, pp. 73-84. ACM, 2006.
- [5] C. Burrell and M. Melchert, “Augmenting compiler error reporting in the Karel++ Microworld,” in *Proceedings of the 20th Annual Conference of the National Advisory Committee on Computing Qualifications (NACCQ '07)*, pp. 41–46, New Zealand, 2007.
- [6] B. S. Lerner, M. Flower, D. Grossman, and C. Chambers, “Searching for type-error messages,” in *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '07)*, pp. 425–434, San Diego, Calif, USA, 2007