RV College of Engineering

IEEE BANGALORE SECTION

FIU FLORIDA INTERNATIONAL UNIVERSITY

Go, change the world

7th IEEE International Conference CSITSS-2023

November 2nd to 4th 2023

Paper ID: 241

Title : Recursive Descent Parser for Abstract Syntax Tree Visualization of Mathematical Expressions

Authors: Shiva Teja Pecheti, Basavadeepthi H M, Nithin Kodurupaka, Meena Belwal

Affiliation: Department of Computer Science and Engineering, Amrita School of Computing,

Bengaluru, India

Outline

- Introduction
- Background Work / Literature Review/ Research Gap
- Objectives & Methodology
- Novelty
- Design & Implementation
- Result Analysis
- Conclusion

Introduction

- Mathematical expression recognition (MER) is the task of finding and evaluating mathematical expressions from different sources, such as text, images, and speech.
- MER is important for many applications in scientific computing, data analysis, engineering simulations, and education.
- Existing methods for MER typically rely on machine learning techniques, such as neural networks and probabilistic context-free grammars.
- These methods require large datasets of labeled examples to train, and they can be computationally expensive.
- Our proposed method for MER uses **Abstract Syntax Trees (ASTs)** and **Recursive Descent Parsing (RDP)**.
- ASTs provide a tree-like representation of the structure and meaning of mathematical expressions.
- RDP is a technique for parsing ASTs by following a set of rules that define the grammar of mathematical expressions.

Literature Review

Author Names	Title	Summary Proposed Method	Required Resources
Zhang, J., Du, J., & Dai, L.	Track, Attend, and Parse (TAP): An End-to-End Framework for Online Handwritten Mathematical Expression Recognition	An end-to-end framework for online handwritten mathematical expression recognition that uses a recurrent neural network and guided hybrid attention.	Large dataset of handwritten mathematical expressions
Noya, E., Benedí, J. M., Sánchez, J. A., & Anitei, D.	Discriminative Learning of Two-Dimensional Probabilistic Context-Free Grammars for Mathematical Expression Recognition and Retrieval	A discriminative learning approach for two-dimensional Probabilistic Context-Free Grammars (PCFGs) for mathematical expression recognition and retrieval.	Large dataset of mathematical expressions
Anitei, D., Sánchez, J. A., & Benedí, J. M.	Py4MER: A CTC-based Mathematical Expression Recognition System	A CTC-based mathematical expression recognition system that uses a Connectionist Temporal Classification (CTC) model.	Large dataset of mathematical expressions
Karamolegkos, P., Kiourtis, A., Karabetian, A., Voulgaris, K., Poulakis, Y., Mavrogiorgou, A., Filippakis, M., & Kyriazis, D.	MathBlock: Performing Complex Mathematical Operations on Synthetic Data	A versatile parser for advanced operations on synthetic data that uses a graph neural network.	Synthetic dataset of mathematical expressions
Menon, V. K., & Soman, K.	A New Evolutionary Parsing Algorithm for LTAG	A new evolutionary parsing algorithm for Linear Tree Adjoining Grammars (LTAG) that improves the efficiency and accuracy of the parsing algorithm.	Large dataset of mathematical expressions
Reddy, B. R., Rup, D. C., Rohith, M., & Belwal, M.	Indian Sign Language Generation from Live Audio or Text for Tamil	A method for generating Indian Sign Language from live audio or text for Tamil.	Large dataset of Tamil text and sign language videos
Ba, A., Lynch, K., Ploennigs, J., Schaper, B., Lohse, C., & Lorenzi, F.	Automated Configuration of Heterogeneous Graph Neural Networks with a Semantic Math Parser for IoT Systems	A method for automating the configuration of heterogeneous graph neural networks for IoT systems using a semantic math parser.	Large dataset of IoT system data
Sowmya, C., Deena, S., & Anbuchelian, S.	Performance of Sentimental Analysis by Studying and Mining Social Media Using Parsing Technique	A method for performing sentimental analysis of social media data using parsing techniques.	Large dataset of social media data

Objectives

- To develop a novel method for mathematical expression recognition (MER) using Abstract Syntax Trees (ASTs) and Recursive Descent Parsing (RDP).
- To demonstrate the accuracy and efficiency of our proposed method on a variety of mathematical expressions.
- To explore the potential of our proposed method in real-world applications.
- Our proposed method for MER uses ASTs and RDP to represent and parse mathematical expressions in a robust and efficient manner.
- Our method can handle complex mathematical expressions, including those with variables and functions

Methodology

Our proposed method for MER consists of the following steps:

- **Lexical analysis**: The input expression is broken down into tokens, which are the basic building blocks of mathematical expressions, such as numbers, operators, and variables.
- **AST construction**: An AST is constructed for the input expression using RDP. The AST represents the structure and meaning of the expression in a tree-like form.
- **Expression evaluation**: The AST is evaluated to calculate the value of the expression. This is done by recursively evaluating the subtrees of the AST.
- **Graph visualization** (optional): For some expressions, such as those involving functions, graphs can be generated to help users understand their behavior and properties.

Novelty

- It uses Abstract Syntax Trees (ASTs) and Recursive Descent Parsing (RDP) to represent and parse mathematical expressions. *This approach is more robust and efficient than existing methods, which typically rely on machine learning techniques.*
- Our method can handle complex mathematical expressions, including those with variables and functions. *This is in contrast to many existing methods, which are limited to simpler expressions.*
- Our method is also more adaptable than existing methods. *It can be easily extended to handle new types of mathematical expressions and to be used in new applications.*

Design and Implementation

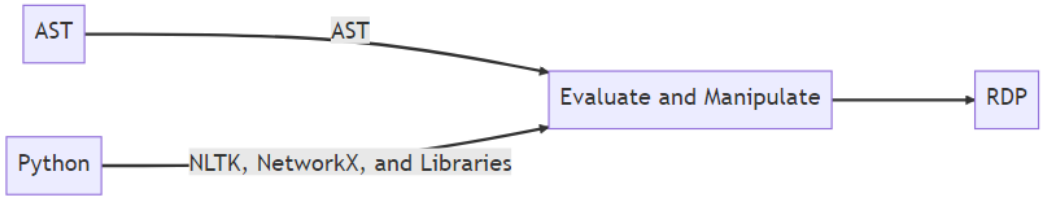
The input expression is broken down into tokens using a regular expression tokenizer.

The AST is evaluated to calculate the value of the expression using the Visitor Pattern.

AST is constructed for the input expression using RDP.

The parser uses the Visitor Pattern to implement different operations on the AST.

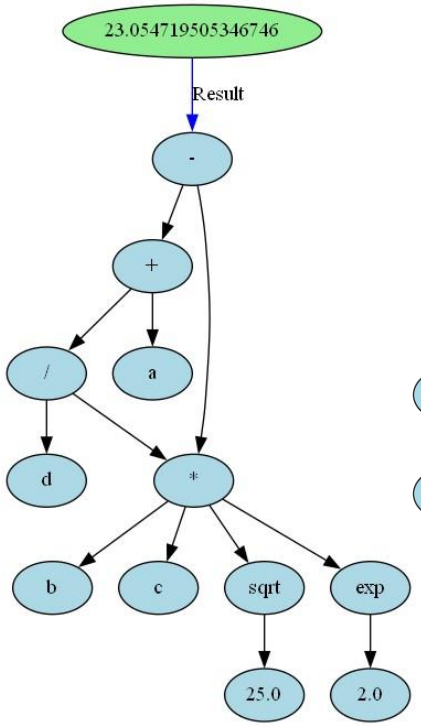
AST can be visualized using the Graphviz library.



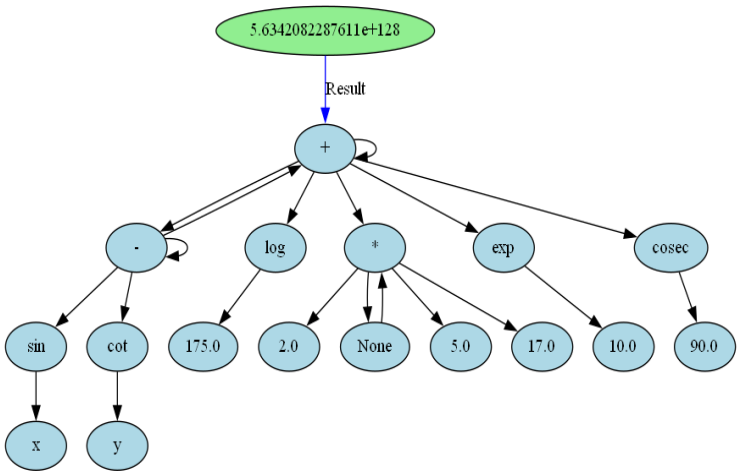
- The parser uses an AST to represent the structure and meaning of mathematical expressions.
- The parser uses RDP to parse complex expressions.

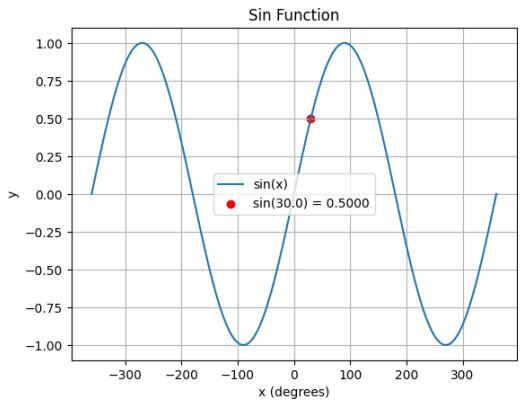
Results and Analysis

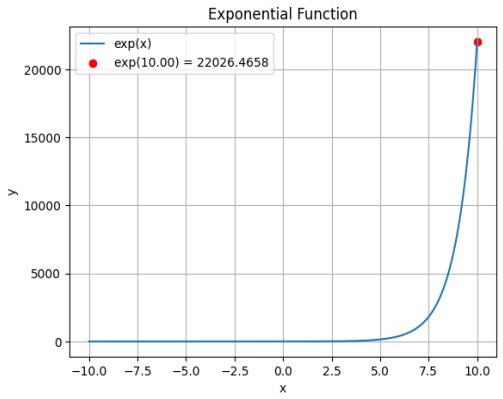
Test Case 1: a + b * (c - d) / e

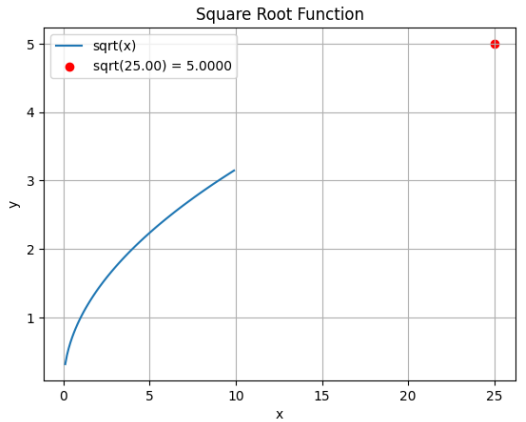


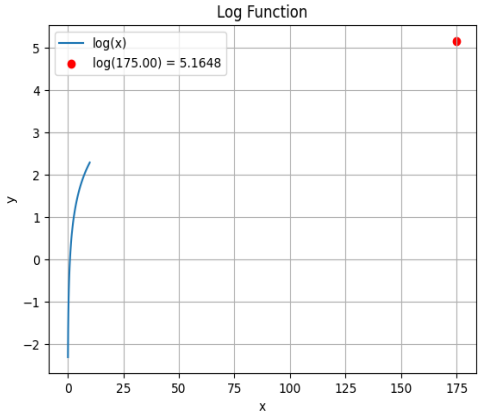
Test case 2 : a + b * c / d - sqrt(25) * exp(2)











- The parser was able to accurately construct the Abstract Syntax Tree (AST) for each test case, handle variable substitution, and evaluate the expression correctly.
- It was also able to visualize the graphs of trigonometric, logarithmic, and exponential functions for better understanding of their behavior.
- These results demonstrate that the mathematical expression parser is a reliable and versatile tool for evaluating and visualizing mathematical expressions.

Potential Applications

- **Education**: The parser can be used to develop educational tools that help students learn about mathematical expressions and functions. *For example, the parser could be used to create a tool that allows students to visualize the graphs of mathematical functions.*
- **Engineering and science**: The parser can be used to develop engineering and scientific applications that require the evaluation of complex mathematical expressions. *For example, the parser could be used to develop a tool that helps engineers to design bridges or airplanes.*
- **Finance and economics**: The parser can be used to develop financial and economic applications that require the evaluation of complex mathematical expressions. *For example, the parser could be used to develop a tool that helps traders to analyze stock prices.*

Conclusion

- The mathematical expression parser is a well-designed and implemented tool for evaluating and visualizing mathematical expressions. It is reliable, versatile, and easy to use.
- The experimental results show that the parser is capable of handling a variety of mathematical expressions, including basic expressions with variables, expressions with functions, and complex expressions with a combination of operations and functions.
- The parser was able to accurately construct the AST, handle variable substitution, and evaluate the expression correctly in all test cases.
- It was also able to visualize the graphs of trigonometric, logarithmic, and exponential functions for better understanding of their behavior.