A Visual Improvement to the Pedagogy of Introductory Logic

Department of Computer Science

Ali Altamimi, Harinderveer Badesha, Christopher Brantley, Joshua Crotts, Nadia Doudou
Spring 2021

Abstract

The Problem:

- Formal logic is difficult for many students to understand at first
- Resources online can conflict - different symbols are used in different courses
- Practice makes perfect, but solutions can be hard to follow
- Proofs and discrete math do not always come naturally to some

The Solution:

- Develop a tool for students that exposes them to propositional and first-order predicate logic
- Emphasis on graphical user interface, an easy-to-use program enhances the learning experience
- The tool should complement the traditional textbook and lecture pedagogy
- Display rules, axioms, and definitions in a comprehensible and legible manner

What is LLAT?

LLAT stands for the Logic-Learning Assistance Tool. Not only are there several features present for visualizing and understanding algorithms, we also support

- Multiple language support
- Different colored themes
- Login/Registration to remotely save preferences for theme, language, and last ten used well-formed formulas
- Exporting workspace to .pdf and .tex (LaTeX)

LLAT also provides the following algorithms for students to use:

- Building and constructing truth tables, parse, and truth trees
- Determine logical relationships between well-formed formulas
- Generate random propositional and predicate logic formulas
- Determine the validity of an argument with truth tree method
- Semantic entailment determiner
- Bound/free variable detector, open/closed/ground sentence determiner
- Main operator detector/finder

More details on the algorithms are offered in the user manual.

Interface Examples

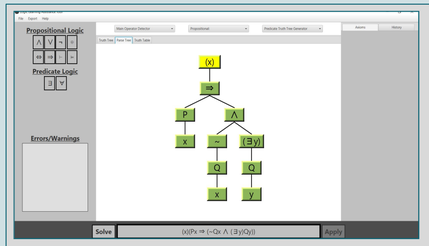

Figure 1: Parse tree of a predicate logic formula.

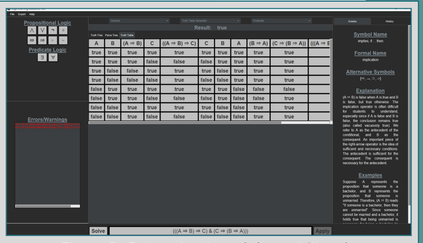

Figure 2: Determination of the validity of a propositional logic formula.

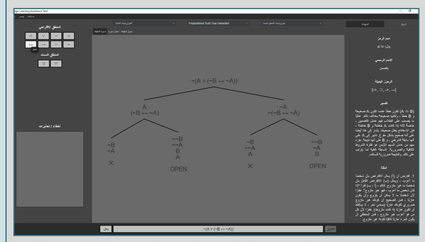

Figure 3: Testing logical equivalence using truth tree, dark theme, Arabic in-progress translation.

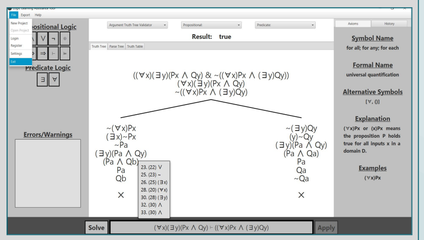

Figure 4: Argument validity check with result at the top in first-order predicate logic.

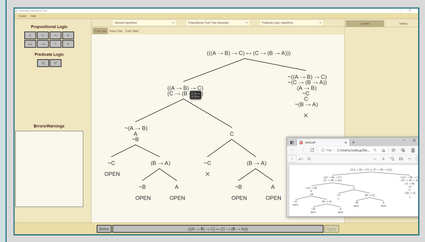

Figure 5: Tan theme example with export to PDF displayed.

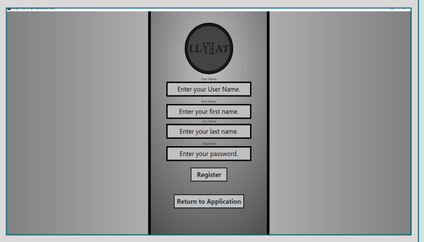

Figure 6: Register screen so users can save preferences (theme, language), and last ten wffs.

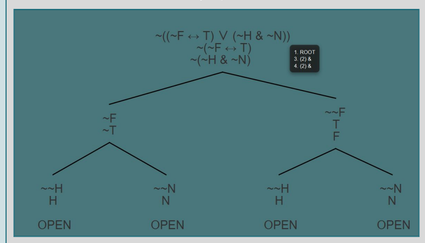

Figure 7: Tooltip over truth tree.

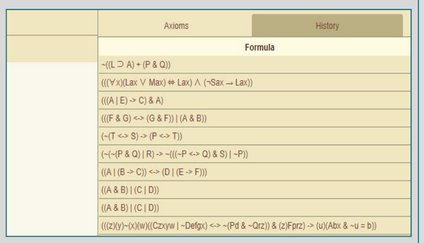

Figure 8: History saved from user's database.

Subsystem

Algorithm: ANTLR is the parsing library which generates the well-formed formula trees.

Database: Stores user information and data in the cloud.

Local Storage: Stores user settings locally for saved preferences on application launch.

Frontend/User-Interface: Graphical user interface generates trees using the Abego Java library.

Language API: Google Script that allows translation of application for all languages.

Future Work

Future Work:

- Add more algorithms: natural deduction is a possibility!
- Improve the performance of the algorithms that we do have
  - Truth tables have a hard restriction on the number of atoms (SAT)
  - Computational complexity is a big limitation
- Add step-by-step directions and instructions for each algorithm

Future Research:

- Improve the random PL and FOPL generation algorithms
  - Practice makes perfect!

Fix Bugs:

- Language translation is hit or miss - possibly translate them by hand?
- Labels not translating correctly
- Sizing of labels and buttons are incorrect for some languages
- Background and border colors conflict with text colors, need to implement a better color management system