ILTIS: Teaching Logic in the Web

Gaetano Geck\textsuperscript{1}, Christine Quenkert\textsuperscript{2}, Marko Schmellenkamp\textsuperscript{3}, Jonas Schmidt\textsuperscript{1}, Felix Tschirbs\textsuperscript{1}, Fabian Vehlken\textsuperscript{1} and Thomas Zeume\textsuperscript{3}

\textsuperscript{1}TU Dortmund
\textsuperscript{2}FH Dortmund
\textsuperscript{3}Ruhr University Bochum

\{gaetano.geck, jonas2.schmidt, felix.tschirbs, fabian.vehlken\}@tu-dortmund.de, quenkertc@gmail.com, \{marko.schmellenkamp, thomas.zeume\}@rub.de

1 Introduction and Motivation

Logical methods are a foundation of many subdisciplines of artificial intelligence. Also other areas of computer science — such as formal software and hardware verification and query languages for databases — rely on logical foundations.

Teaching the foundations of logics and, more generally, of formal methods to the next generation of computer scientists is, for these reasons, integrated into recommendations for Bachelor curricula (see, e.g., [ACM and IEEE, 2013; Gesellschaft für Informatik, 2016]). However, the rapid increase of the number of students enrolled in computer science courses over the last ten years [Computing Research Association, 2017] is problematic for instructors in at least two ways.

On the one hand, students start with diverse backgrounds in basic STEM education, so that many of them often struggle with the foundations of formal methods. On the other hand, individual, human tutoring for so many students is a challenge. This is in particular true for MOOCs (massive open online courses), which are becoming more and more popular, but generally suffer from high drop-out rates ([Shah, 2020], see also, e.g., [Siemens, 2013]).

Improving motivation and understanding in formal methods education is, therefore, an important goal. Also the National Research Council of the US advocates, among others, to “leverage technologies to make the most effective use of students’ time, shifting from information delivery to sense-making and practice in class” (see [Singer et al., 2012], summary from [Beach et al., 2012]).

The ILTIS teaching support system aims at implementing this objective for the foundations of formal methods [Geck et al., 2018; Geck et al., 2019]. It offers a flexible, modular framework for designing and combining educational tasks. A variety of educational tasks for typical pipelines of applying logical methods have already been implemented (see Section 2 for more details). For instance, automated reasoning pipelines — modelling scenarios with logical formulas, transforming formulas into suitable normal forms, and inferring new knowledge — for propositional logic, modal logic, as well as first-order logic are covered to a large extent. For each task, feedback can be declaratively specified by combining predefined feedback generators (see Section 3).

The goal of this demonstration is to announce that the ILTIS teaching support system has reached maturity for use in introductory courses for formal methods.

As a proof-of-concept, extensive interactive material has been designed for an introductory course “Logic for computer scientists”, attended by more than 200 students at TU Dortmund University in the winter term 2020/2021. This includes material for practicing concepts, multiple choice quizzes, and interactive tutorials throughout the course.

Since the last report on the ILTIS system in [Geck et al., 2018; Geck et al., 2019], the focus has been on improving usability of the system and extending its portfolio of educational tasks. For instance:

\begin{itemize}
  \item The graphical user interface has been re-designed, taking didactical and usability aspects into account.
  \item A core aspect of ILTIS is that instructors can combine educational tasks into more complex exercises. For more flexibility, new logic-related tasks (e.g. inference for modal logic and model construction for first-order logic) as well as transitional tasks have been added.
  \item The feedback system has been re-designed to allow instructors to specify how feedback is provided in a more flexible, yet simple declarative language.
\end{itemize}

Related work In the past, a large variety of educational tools for teaching the foundations of formal methods have been developed [Antonia and Sánchez, 2011]. Many of them

\textsuperscript{1}Try it: https://iltis.cs.tu-dortmund.de
are not web-based and thus have a high entry-barrier. There are a few modern, web-based tools.

For teaching the foundations of logic. The LogEX system supports transformation of propositional formulas [Lodder et al., 2015; Lodder and Heeren, 2011]. The focus of the Logic4Fun project is on modeling scenarios and puzzles by logical means [Slaney, 2015]. AELL [Huertas et al., 2011] offers tasks for natural deduction, truth tables, and resolution but is not publicly available. Modal logic semantics can be explored with Hintikka’s World [Charrier et al., 2019].

For teaching the foundations of formal languages. Constructing models for formal languages such as regular expressions, automata, and formal grammars can be practiced with a variety of tools. In FLACI [Hielscher and Wagenknecht, 2019], students can convert between these models. The AutomataTutor [Alur et al., 2013; D’Antoni et al., 2015a; D’Antoni et al., 2015b; D’Antoni et al., 2020] offers elaborate feedback on finite automata construction tasks. JFLAP [Gramond and Rodger, 1999; Rodger, 1999] features many algorithms on automata, but is not web-based.

2 Designing content in ILTIS

Exercises in ILTIS are built from small, easily composable, educational tasks. Each educational task is configured by inputs — either given explicitly or as the output of prior tasks — and provides outputs for the objects created by students. The outputs can then be used by subsequent educational tasks. For instance, a task for transforming a formula into conjunctive normal form (CNF) gets a formula as input and provides the student-constructed, equivalent formula in CNF as output. The structure of a typical multi-step exercise — the complete reasoning workflow — is provided in Figure 1 (instantiations can be found in the course material).

Currently, ILTIS supports a variety of educational tasks for propositional logic, modal logic, and first-order logic (see Table 1 for a high-level summary):

- **Evaluating formulas**: Students can evaluate formulas for a given interpretation. For propositional logic, students can construct truth tables; for modal logic, students can construct evaluation tables for a given Kripke structure.
- **Constructing models**: Students can construct models (i.e., satisfying interpretations) for formulas. For propositional logic, models are specified by valuations of all variables; for modal logic by Kripke structures. For first-order logic, students can construct models for formulas describing properties of (colored) graphs.
- **Constructing formulas**: Students can construct formulas for the respective logics. For propositional and modal logic, formulas are checked for correctness and, if they are not, underlying misconceptions are determined and used to provide adequate feedback (see Section 3). For first-order logic, another approach is taken due to algorithmic intractability of testing equivalence. Here, instructors provide a textual description of a property of nodes in a graph as well as a sample graph and ask students to construct a first-order formula with one free variable that selects the nodes which satisfy the property.
- **Applying equivalence transformations**: Students can transform formulas step-by-step into an equivalent target formula or into an equivalent formula in a normal form such as conjunctive, disjunctive, or negation normal form. Currently, this educational task is available for propositional and modal logic.
- **Testing satisfiability**: Students can test formulas for satisfiability using several methods. For propositional logic, students can a) construct truth tables, b) construct tableaux according to the tableau calculus, c) apply the algorithm for testing satisfiability of Horn formulas, or d) construct a resolution graph. For modal logic, students can apply the tableau calculus. For first-order logic, students can construct a resolution graph.

Further educational tasks for 1) smoothing complex exercise workflows (e.g., multiple choice tasks) and for 2) specific content (such as bisimulation, providing witnesses for non-equivalence, etc.) are available.

3 Feedback in ILTIS

Feedback is one of the most important factors for learning success. It is therefore one of the core objectives of ILTIS to provide immediate and comprehensive feedback.
In the feedback framework of ILTIS, each educational task type comes with (possibly) multiple feedback generators, each one responsible for one kind of feedback. Individual feedback generators can be composed to feedback strategies by simple rule-based programs. Such programs determine the order of feedback and the invocation of rules may depend on the success of previous rules.

When specifying interactive exercises, instructors can state which feedback strategy to use (or define a custom one). In this way, the progress of students can be taken into account, e.g., a strategy that provides a lot of feedback can be used for beginners, while a strategy that provides almost no feedback can be used for exam preparation. As an example, each section in the feedback box (indicated by thin red lines) in Figure 2 is provided by a different feedback generator.

Designing feedback generators is a subtle and challenging task, in particular as algorithmic feasibility has to be taken into account. To provide an idea, we sketch two feedback generators currently used in ILTIS:

- **Identifying misconceptions for constructing propositional and modal formulas.** Using sample data from exams, we extracted typical mistakes made by students when constructing propositional and modal formulas. These include, for example, mixing up the premise and the conclusion of an implication (i.e. “if” and “only if”) or confusing the box and the diamond operators in modal logic. Each identified type of mistake can be captured by a general transformation rule and be used to generate feedback as provided in Figure 2. For example, the transformation rule $\square X \sim \Diamond X$ describes that a student erroneously used a box instead of a diamond operator. While in [Geck et al., 2018] this mechanism was solely used for propositional logic, it is now also supported for modal logic and a variant for first-order logic is in development.

- **Visual feedback for constructing first-order formulas.** As described above, ILTIS currently supports the construction of first-order formulas only for graph formulas with a single free variable, selecting nodes satisfying a given property. Visual feedback is provided by evaluating a student-provided formula and comparing the result to the nodes selected by a correct formula (see Figure 3).

4 Future work and (shameless) call for help

We are looking for:

- Instructors who want to use ILTIS in their courses and thereby provide helpful feedback for future directions.
- Interested students, PhD students, and PostDocs for helping us to a) lay the theoretical foundations for ILTIS (e.g. the development of suitable feedback mechanisms in algorithmically hard domains) and to b) drive forward the coverage of topics by ILTIS.
- Experts in natural language processing for help in building educational tasks for bridging the gap between natural languages and formal modeling.

Acknowledgments

Many people are contributing to the ILTIS project in various ways. Without them, the system would not exist in its current form. The implementation of the system has been supported by Artur Ljulin, Johannes May, Sebastian Peter, Patrick Roy, and Daniel Sonnabend. The creation of the course material has been supported by Jill Berg, Alicia Gayda, Melanie Schwartz, and Cara Volbracht. Thomas Schwentick and Christopher Spinrath are providing advice for many aspects of the project.

We acknowledge the financial support by DFG grant ZE 1235/2-1.
References


