

ILTIS: Teaching Logic in the Web

Gaetano Geck¹, Christine Quenkert², Marko Schmellenkamp³, Jonas Schmidt¹,
Felix Tschirbs¹, Fabian Vehlken¹ and Thomas Zeume³

¹TU Dortmund

²FH Dortmund

³Ruhr University Bochum

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

Abstract

The ILTIS project provides an interactive, web-based system for teaching the foundations of formal methods. It is designed to allow modular addition of educational tasks as well as to provide immediate and comprehensive feedback. Currently, exercises for various aspects of typical automated reasoning workflows for propositional logic, modal logic, and first-order logic are covered.

Recently, ILTIS has reached a level of maturity where large parts of introductory logic courses can be supplemented with interactive exercises. Sample interactive course material has been designed and used in such courses with more than 200 students.

We invite all readers to give it a try!¹

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:

- The graphical user interface has been re-designed, taking didactical and usability aspects into account.
- 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.
- The feedback system has been re-designed to allow instructors to specify how feedback is provided in a more flexible, yet simple declarative language.

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

¹Try it: <https://iltis.cs.tu-dortmund.de>

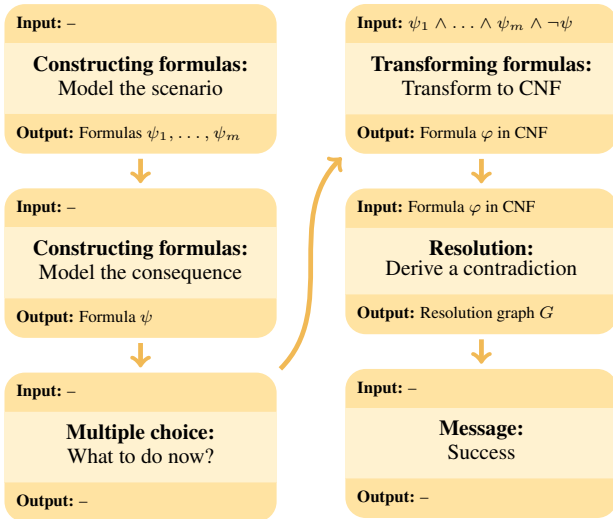


Figure 1: A complex exercise for reasoning about a real world scenario built from smaller educational tasks. Each task is accompanied by natural language descriptions. The output of the tasks is provided by students in the web-frontend.

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 Ta-

| Task | Propositional logic | Modal logic | First-order logic |
|------------------------|---------------------|-------------|-------------------|
| Evaluating formulas | ✓ | ✓ | ⚙️ |
| Constructing models | ✓ | ✓ | (✓) |
| Constructing formulas | ✓ | ✓ | (✓) |
| Transforming | ✓ | ✓ | ⚙️ |
| Testing satisfiability | ✓ | ✓ | (✓) |

Table 1: A summary of tasks that are supported (✓), supported with limitations ((✓)), and in development (⚙️).

ble 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 untractability 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 either 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.

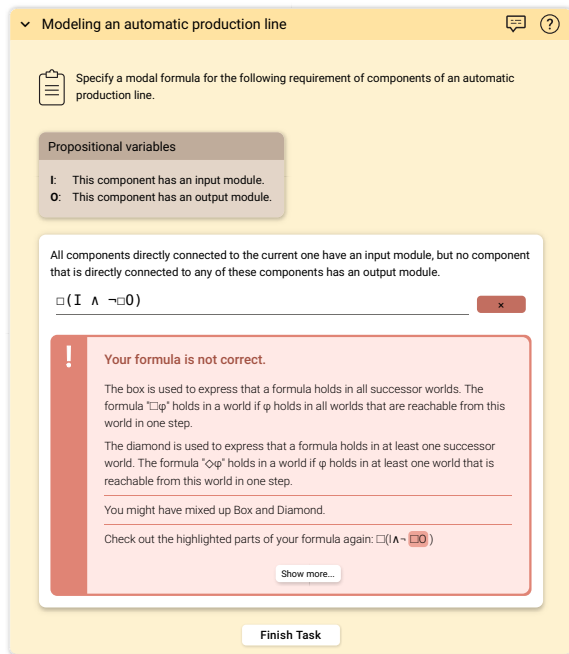


Figure 2: An example task for constructing modal formulas.

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 $\Box X \rightsquigarrow \Diamond X$ 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 sup-

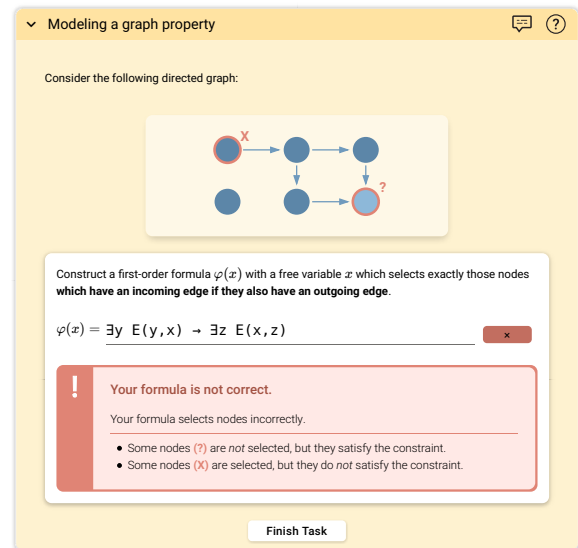


Figure 3: An example task for constructing first-order formulas.

ported 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

- [ACM and IEEE, 2013] Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. Association for Computing Machinery, New York, NY, USA, 2013.
- [Alur *et al.*, 2013] Rajeev Alur, Loris D’Antoni, Sumit Gulwani, Dileep Kini, and Mahesh Viswanathan. Automated grading of DFA constructions. In Francesca Rossi, editor, *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 1976–1982. IJCAI/AAAI, 2013.
- [Antonia and Sánchez, 2011] María Antonia and Huertas Sánchez. A classification of tools for learning logic. <http://hdl.handle.net/10609/6501>, 2011. Accessed: 2021-03-09.
- [Beach *et al.*, 2012] Andrea L. Beach, Charles Henderson, and Noah Finkelstein. Facilitating change in undergraduate stem education. *Change: The Magazine of Higher Learning*, 44(6):52–59, 2012.
- [Charrier *et al.*, 2019] Tristan Charrier, Sébastien Gamblin, Alexandre Niveau, and François Schwarzentruher. Hintikka’s world: Scalable higher-order knowledge. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 6494–6496. ijcai.org, 2019.
- [Computing Research Association, 2017] Computing Research Association. Generation CS: Computer science undergraduate enrollments surge since 2006. <http://cra.org/data/generation-cs>, 2017. Accessed: 2021-03-09.
- [D’Antoni *et al.*, 2015a] Loris D’Antoni, Dileep Kini, Rajeev Alur, Sumit Gulwani, Mahesh Viswanathan, and Björn Hartmann. How can automatic feedback help students construct automata? *ACM Transactions on Computer-Human Interaction*, 22(2):pp. 9:1–9:24, 2015.
- [D’Antoni *et al.*, 2015b] Loris D’Antoni, Matthew Weavery, Alexander Weinert, and Rajeev Alur. Automata tutor and what we learned from building an online teaching tool. *Bulletin of the EATCS*, 117, 2015.
- [D’Antoni *et al.*, 2020] Loris D’Antoni, Martin Helfrich, Jan Křetínský, Emanuel Ramneantu, and Maximilian Weininger. Automata tutor v3. In Shuvendu K. Lahiri and Chao Wang, editors, *Computer Aided Verification – 32nd International Conference, CAV 2020, Proceedings, Part II*, volume 12225 of *Lecture Notes in Computer Science*, pages 3–14. Springer, 2020.
- [Geck *et al.*, 2018] Gaetano Geck, Artur Ljulin, Sebastian Peter, Jonas Schmidt, Fabian Vehlken, and Thomas Zeume. Introduction to Iltis: an interactive, web-based system for teaching logic. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2018*, pages 141–146. ACM, 2018.
- [Geck *et al.*, 2019] Gaetano Geck, Artur Ljulin, Jonas Haldimann, Johannes May, Jonas Schmidt, Marko Schmellenkamp, Daniel Sonnabend, Felix Tschirbs, Fabian Vehlken, and Thomas Zeume. Teaching logic with Iltis: an interactive, web-based system. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, page 307. ACM, 2019.
- [Gesellschaft für Informatik, 2016] Gesellschaft für Informatik e.V. Empfehlungen für Bachelor- und Master-Programme im Studienfach Informatik an Hochschulen. <https://dl.gi.de/handle/20.500.12116/2351>, 2016. Accessed: 2021-03-09.
- [Gramond and Rodger, 1999] Eric Gramond and Susan H. Rodger. Using JFLAP to interact with theorems in automata theory. In Jane Prey and Robert E. Noonan, editors, *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education, 1999*, pages 336–340. ACM, 1999.
- [Hielscher and Wagenknecht, 2019] Michael Hielscher and Christian Wagenknecht. FLACI – eine Lernumgebung für theoretische Informatik. In Arno Pasternak, editor, *Informatik für alle, 18. GI-Fachtagung Informatik und Schule, INFOS 2019*, volume P-288 of *LNI*, pages 211–220. Gesellschaft für Informatik, 2019.
- [Huertas *et al.*, 2011] Antonia Huertas, Josep M. Humet, Laura López, and Enric Mor. The SELL project: A learning tool for e-learning logic. In Patrick Blackburn, Hans van Ditmarsch, María Manzano, and Fernando Soler-Toscano, editors, *Tools for Teaching Logic - Third International Congress, TICTTL 2011. Proceedings*, volume 6680 of *Lecture Notes in Computer Science*, pages 123–130. Springer, 2011.
- [Lodder and Heeren, 2011] Josje Lodder and Bastiaan Heeren. A teaching tool for proving equivalences between logical formulae. In *Tools for Teaching Logic*, pages 154–161, 2011.
- [Lodder *et al.*, 2015] Josje Lodder, Bastiaan Heeren, and Johan Jeuring. A pilot study of the use of logex, lessons learned. *CoRR*, abs/1507.03671, 2015.
- [Rodger, 1999] Susan H. Rodger. Teaching automata theory with JFLAP. *SIGACT News*, 30(4):53–56, 1999.
- [Shah, 2020] Dhawal Shah. By the numbers: Moocs in 2020. <https://www.classcentral.com/report/mooc-stats-2020>, 2020. Accessed: 2021-03-09.
- [Siemens, 2013] George Siemens. Massive open online courses: Innovation in education? In Rory McGreal, Wanjira Kinuthia, Stewart Marshall, and Tim McNamara, editors, *Open Educational Resources: Innovation, Research and Practice*, pages 5–15. Commonwealth of Learning and Athabasca University, Vancouver, British Columbia and Edmonton, Alberta, 2013.
- [Singer *et al.*, 2012] Susan R. Singer, Natalie R. Nielsen, and Heidi A. Schweingruber. Discipline-based education research. *Washington, DC: The National Academies*, 2012.
- [Slaney, 2015] John Slaney. Logic considered fun. *CoRR*, abs/1507.03683, 2015.