Designing human-like automated assessment to replace proportional penalties for error types

Rein Prank
University of Tartu, Estonia
rein.prank@ut.ee

Types of exercises

We consider two kinds of algebraic exercises in Basic course of Mathematical Logic:

1) Truth-table exercises (filling the truth-table, checking of tautologicity, satisfiability, equivalence and inference),
2) Formula manipulation exercises (expression using $\{\neg, \&\}$, $\{\neg, \vee\}$, $\{\neg, \supset\}$, disjunctive and conjunctive normal forms.

Starting from 1991-92, our students have solved these exercises in computerized environment

Logic or algebra?

Our considerations do not have any specifics of mathematical Logic – really this is pure algebra.

Counting of a row in truth-table is the same as counting of the value of numeric expression in Grade 5
(order of operations, calculation errors)

Transformation to disjunctive normal form is Boolean analog of multiplication of polynomials in Grade 9
(syntax errors, understanding of order of operations, equivalence with previous line, recognising the final form, economy of solutions)

Main features of our exercise environments

The student solves the task step by step

The programs check each step in the solution:

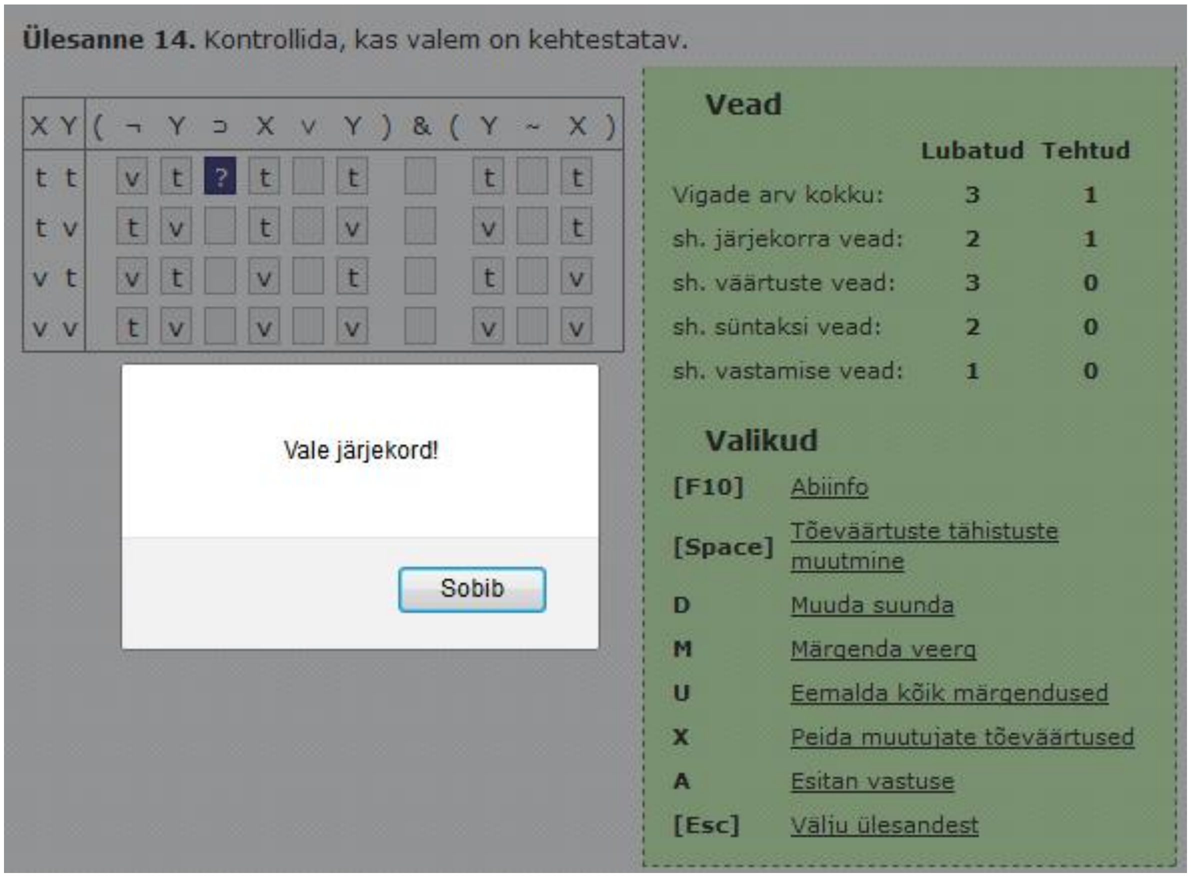
- gives error messages,
- requires correction of errors before the next step.

The programs diagnose and count errors in

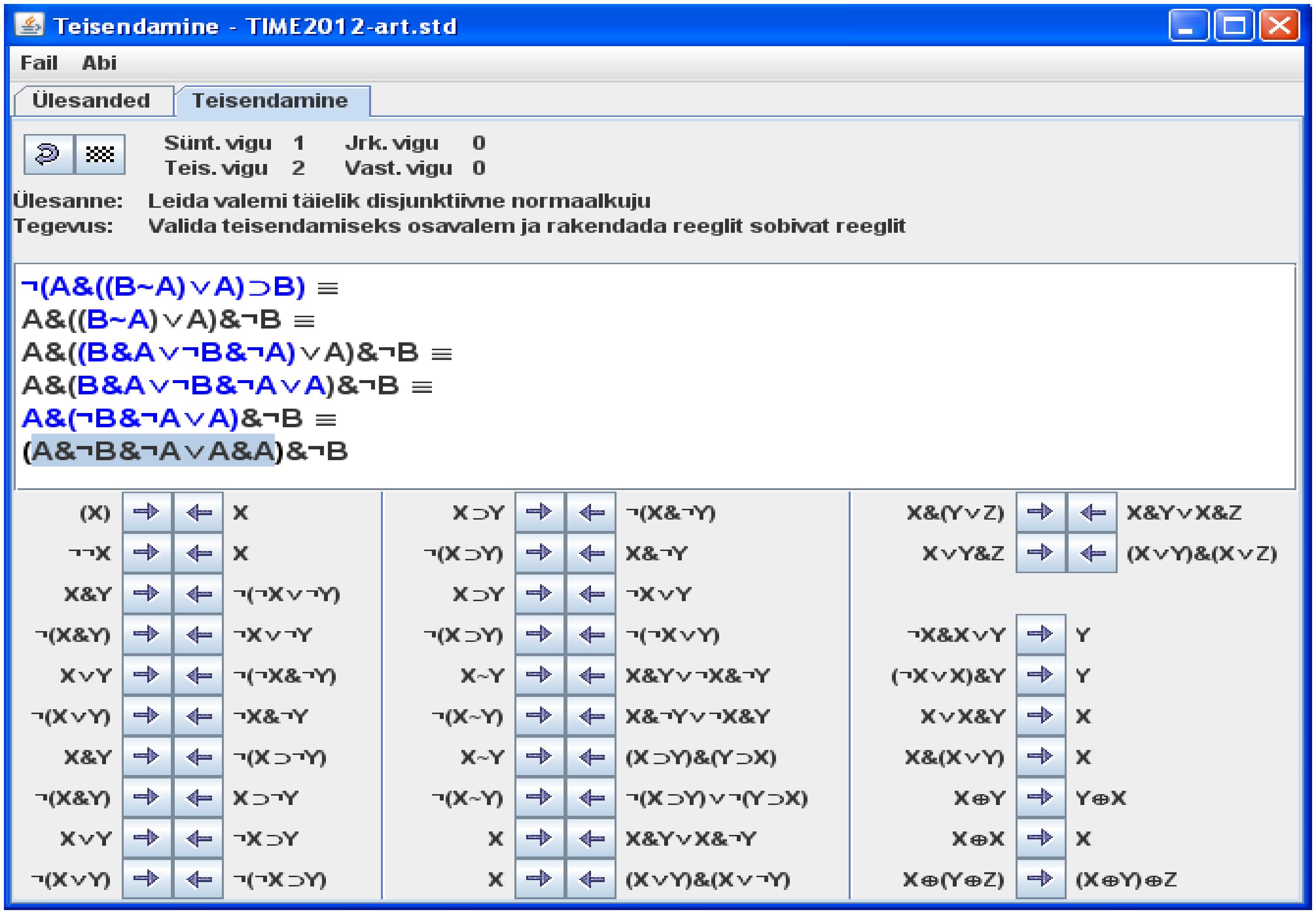
- order of operations,
- truth-value/equivalence,
- syntax,
- answer dialog.

The truth-table environment also enables to establish the penalty for each type of error and counts then the points automatically.

Solution window of truth-table environment (checking of satisfyability)



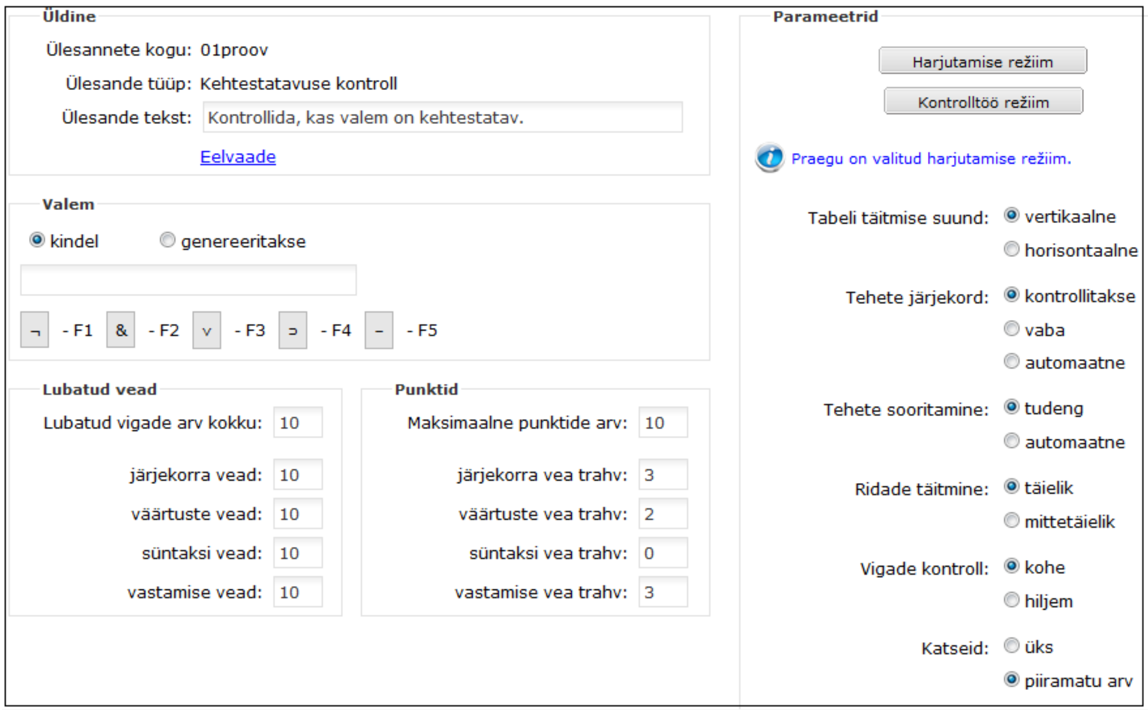
Solution window of formula manipulation environment



1) the student **marks a subformula** to be changed.

2) In the **INPUT mode** the program opens an input box and the student enters a subformula that replaces the marked part.
In the **RULE mode** the student selects a rule from the menu and the program applies it.

Task attributes in truth-table environment. Automated grading (grade = max – penalties)



Formula:
fixed/random

Errors: order, value, syntax, answer

Options:
Direction (horizontal/vertical), *Order* (checked/free/autom), *Operations* (student/autom), *Filling of rows* (full/partial), *Checking* (immediate/before answer), *Attempts* (one/unlimited)

Formalizable aspects of human grading

By human grading we try to take into account at least the following aspects:

1. What part of the task is solved (if the solution is incomplete),
2. Errors,
3. Solution economy/conformity with the algorithm.

The general formula is:
grade =

grade for solved part – error penalties – economy/conformity penalties

In principle, the economy component can be positive in case of a clever nonstandard solution

Framework for formal grading of solved part

1. **The solution algorithm is divided into stages** and completing of each stage gives corresponding percent of points.
2. **The teacher assigns the percents for the stages** (when enters the task using teacher program)**.**
   The alternative is solving the task by automated solver and assigning the percents to the stages according the numbers of necessary steps for actual task. Thereby different types of steps can have different weights.

Grading of solved part (truth-table exercises)

For example in tasks on **checking of tautologicity** division into stages can be:

1) Filling of one arbitrary row : 20%
2) Filling of the table in amount that is sufficient for giving the answer for actual formula and switching to answer dialog : 70%
3) Completed solution: 100%

The same can be done for **other types** where filling of the table is followed by the answer dialog.

Grading of solved part (formula manipulation exercises)

- In case of expression using $\{\neg, \&\}$, $\{\neg, \vee\}$, $\{\neg, \supset\}$ we can assign percents to elimination of each of the three other connectives.
- In case of normal form exercises we can assign percents to each of the completed stages 1-6 in the full normal form algorithm:
  1. Eliminate implications and biconditionals from the formula.
  2. Move negations inside.
  3. Use distributive law to expand the conjunctions of disjunctions.
  4. Exclude contradictory conjunctions and redundant copies of literals.
  5. Add missing variables to conjunctions.
  6. Order the variables alphabetically, exclude double conjunctions.

Refining the stage-based splitting of the work

- Further, in many task types some **stages can be split into some number of steps** that could give proportional parts of the grade assigned to the whole solution stage.
  For example, we can split
  filling of full truth-table into filling of rows,
  elimination of all conjunction symbols into some number of individual eliminations
- **Number of steps depends on the concrete task**
- In some cases this **splitting can be nontrivial**. For example, elimination of biconditionals creates additonal conjunctions in the formula.
- Described here splitting mechanisms should be implemented in the program (using possibly the automated solver) and probably can controlled from the teacher interface only by switching on/off.

Framework for counting the penalty for errors

1. We suppose that **the program diagnoses some types of errors and counts the errors** for each type.
2. **The teacher assigns basic penalty for each type** (in percents or in absolute points) for each task.
3. The teacher defines **penalty calculation function**

Some error penalty calculation principles

By human grading of our computerized tests **we have used:**

1. If the task is solved then give at least 30% of points.
2. Leave one (two) error(s) not penalized.
3. Use sublinear penalty function, for example
   p(1)=0.2, p(2)=0.35, p(3)=0.5, p(4)=p(5)=0.6, p(6)=0.7.

Such principles can be entered using spreadsheet-like functions or as values of suitable parameters.

Grading of global economy (formula manipulation tasks)

- If the student solves the task using ‘RULE MODE’ (predefined types of steps) then we can count the **length of the solution.**
- Thereby different RULES can have **different weights**
- Global economy can be graded comparing
  the **length of student solution**
  with **solution of automated solver** or with **number(s) entered by the teacher**.
- Automated solver enables to grade tasks containing **randomly generated expressions**

Grading of global economy (truth-table tasks)

- If the formula is not a tautology then it is sufficient to calculate only one row in the table

Grading of local economy

The program should evaluate each solution step

- **We can compare** it with
  the step of automated solver or
  with the solution stage defined by the solution algorithm
  and diagnose corresponding mistakes

For example, we have supplementary program for formula manipulation exercises that diagnoses 24 types of algorithmic mistakes.

- Diagnostics is easier to implement for rule-based working modes
- It is necessary to accept the simplification steps/rules

The teacher can assign penalties for error types and enter the penalty calculation function

What should the program be able to do?

1. For grading of solved part: to recognize what stages of the algorithm are completed (often possible without automated solver).
2. For error penalties: to diagnose some number of different error types.
3. For grading of global economy: to count length of the solution, to find ‘standard’ solution
4. For grading of local economy: to diagnose differences with standard algorithm