Universiteit Utrecht

Hints and feedback in learning environments for algebra

technology for following and helping students

Johan Jeuring

Joint work with Alex Gerdes and Bastiaan Heeren

Utrecht University and Open Universiteit Nederland
Computer Science

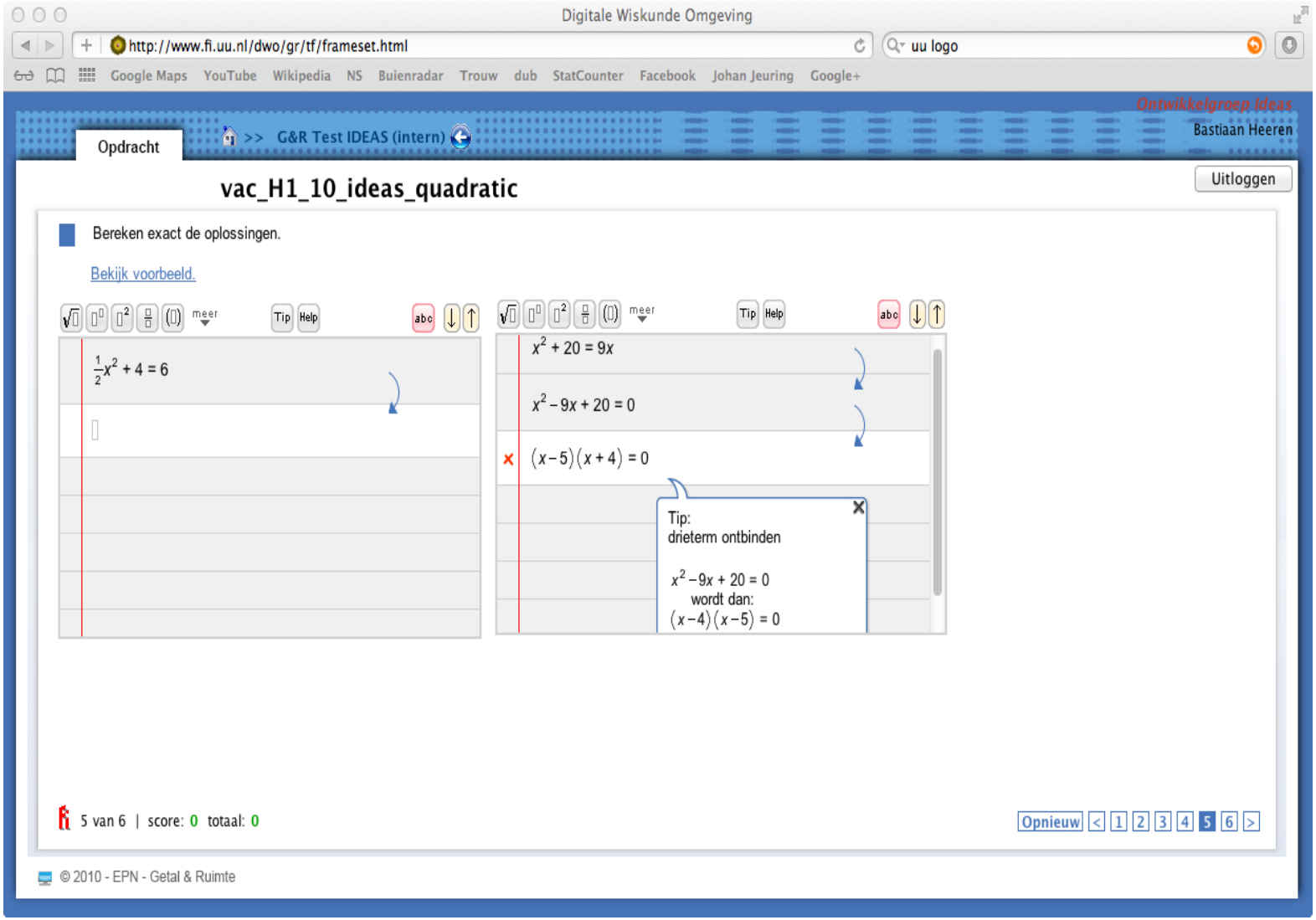
Symposium on Algebra and ICT, Utrecht

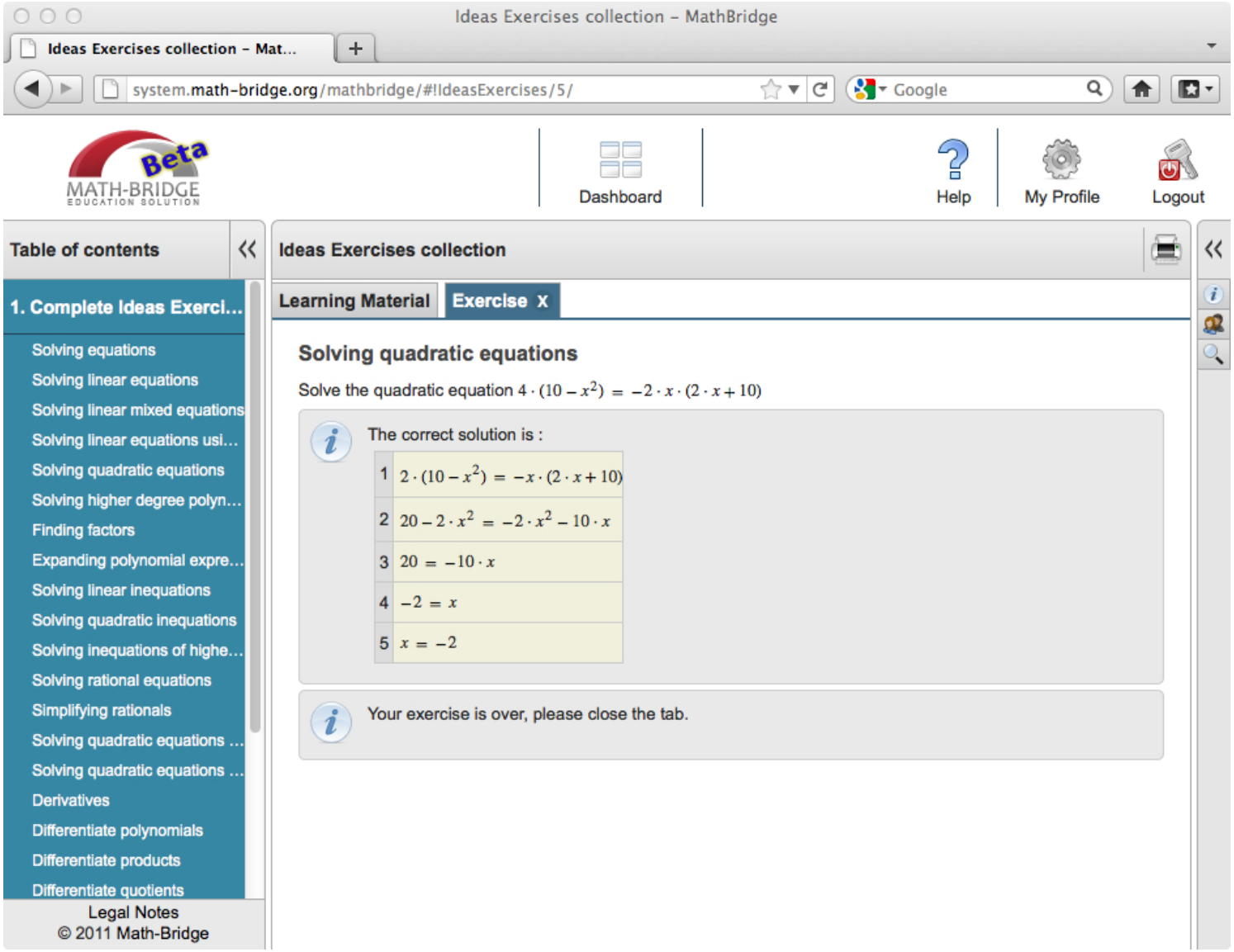
December 13, 2011

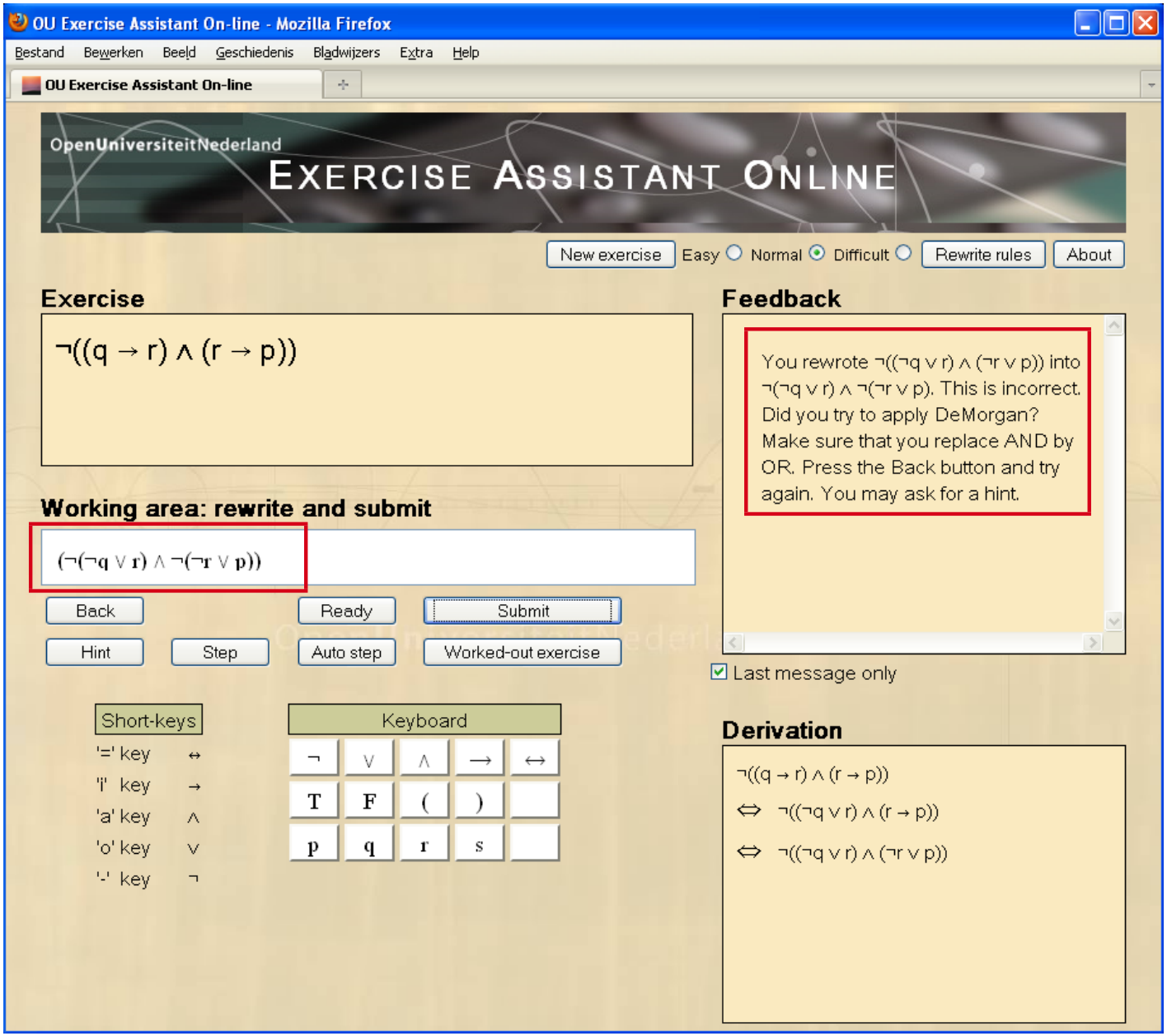
Open Universiteit
www.ou.nl

Learning environments for algebra

- There are many learning environments for algebra
- Learning by doing: learning environments for algebra typically offer a wide variety of interactive exercises
- Learning through feedback: feedback services are often delegated to external tools, such as computer algebra systems, or domain reasoners
- A domain reasoner offers the possibility to:
  - generate steps
  - analyse steps


Digitale Wiskunde Omgeving
http://www.fi.uu.nl/dwo/gr/tf/frameset.html
uu logo
Google Maps YouTube Wikipedia NS Buienradar Trouw dub StatCounter Facebook Johan Jeuring Google+
Ontwikkelgroep Ideas
Opdracht
>> G&R Test IDEAS (intern)
Bastiaan Heeren
Uitloggen
vac_H1_10_ideas_quadratic
Bereken exact de oplossingen.
Bekijk voorbeeld.
meer
Tip
Help
abc
$\frac{1}{2}x^2 + 4 = 6$
$x^2 + 20 = 9x$
$x^2 - 9x + 20 = 0$
$(x-5)(x+4) = 0$
Tip:
drieterm ontbinden
$x^2 - 9x + 20 = 0$
wordt dan:
$(x-4)(x-5) = 0$
5 van 6 | score: 0 totaal: 0
Opnieuw < 1 2 3 4 5 6 >
© 2010 - EPN - Getal & Ruimte


Ideas Exercises collection – MathBridge
Ideas Exercises collection – Mat...
system.math-bridge.org/mathbridge/#!IdeasExercises/5/
Google
Beta
MATH-BRIDGE
EDUCATION SOLUTION
Dashboard
Help
My Profile
Logout
Table of contents
1. Complete Ideas Exerci...
Solving equations
Solving linear equations
Solving linear mixed equations
Solving linear equations usi...
Solving quadratic equations
Solving higher degree polyn...
Finding factors
Expanding polynomial expre...
Solving linear inequations
Solving quadratic inequations
Solving inequations of highe...
Solving rational equations
Simplifying rationals
Solving quadratic equations ...
Solving quadratic equations ...
Derivatives
Differentiate polynomials
Differentiate products
Differentiate quotients
Legal Notes
© 2011 Math-Bridge
Ideas Exercises collection
Learning Material
Exercise X
Solving quadratic equations
Solve the quadratic equation $4 \cdot (10 - x^2) = -2 \cdot x \cdot (2 \cdot x + 10)$
The correct solution is :
1 $2 \cdot (10 - x^2) = -x \cdot (2 \cdot x + 10)$
2 $20 - 2 \cdot x^2 = -2 \cdot x^2 - 10 \cdot x$
3 $20 = -10 \cdot x$
4 $-2 = x$
5 $x = -2$
Your exercise is over, please close the tab.



Universiteit Utrecht

Feedback services

Our domain reasoners offer several feedback services:

- Diagnosis of a step by a user
- Hint on how to continue
- Worked-out solution
- Report common mistake (buggy rule)
- Check finished

This talk:

- How do we provide feedback services?
- What are the important concepts that play a role?

Outline of presentation

1. Introduction
2. Feedback services
3. Rewrite rules
4. Views
5. Strategies
6. Related work
7. Future work and conclusions

Universiteit Utrecht

Outline of presentation

§2

Universiteit Utrecht

Feedback services

A domain reasoner offers the possibility to:

- generate steps
- analyse steps

We generate steps to

- give one or more hints on how to take a next step
- show a worked-out solution

We analyse steps to

- determine the correctness of a step
- detect applications of buggy rules
- check if an exercise is finished

Feedback services via the web

Our feedback services are available via the web, as web services.

For example, a learning environment can ask our domain reasoners to get the first step in solving the linear equation

$$5 \cdot (x - 1) = \frac{x - 1}{2}$$

The tool responds with: remove the division by multiplying with 2. (All in XML.)

Documentation

The documentation on

`http://ideas.cs.uu.nl/docs/latest/exercises.html`

shows the details:

- Services, for example `onefirst`
- Exercises, for example `algebra.equations.linear`
- Rules, for example
  - good: `algebra.equations.linear.remove-div`
  - buggy: `algebra.equations.buggy.minus-minus`
- Examples, see page

Users

The following learning environments use our feedback services:

- The Digital Mathematical Environment from the Freudenthal Institute
  $>$ 100.000 hits
- Math-Bridge (based on ActiveMath)
  $>$ 50.000 hits, but partly untraceable: local installations of our services
- Mathdox
  Couple of thousands hits
- Our own tutor for logic
  $>$ 10.000 hits

Requirements

We have the following requirements for our feedback services, and hence our domain reasoners:

- they can automatically diagnose student interactions
- they satisfy the cognitive fidelity principle
- their components are
  - observable
  - adaptable
  - composable

Furthermore, to make the code-base manageable:

- they are generic

Concepts in our domain reasoners

1. Rewrite rules
   - Specify how terms can be manipulated
   - Can represent common misconceptions ('buggy rules')
2. Views and canonical forms
   - For recognizing forms and defining notational conventions
   - Missing link between rules and strategies
   - Examples: $ax^2 + bx + c = 0$; $3\frac{1}{2}$; $e_1 + e_2 + \ldots + e_n$
3. Rewrite strategies
   - Guide the process of applying rewrite rules
   - Use a strategy language, similar to tactic languages (theorem proving), CFG's, and parser combinator libraries

Outline of presentation

§3

Universiteit Utrecht

Rewrite rules

- `distr-times` rewrite rule for mathematical expressions:

$$a \cdot (b + c) \Rightarrow a \cdot b + a \cdot c$$

- Exercise: solve $3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14$

Rewrite rules

- `distr-times` rewrite rule for mathematical expressions:

$$a \cdot (b + c) \Rightarrow a \cdot b + a \cdot c$$

- Exercise: solve $3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14$

$$\begin{array}{ll} & 3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \\ \Rightarrow & \{ \text{ distr-times } \} \\ & 12 \cdot x = 7 \cdot x - 14 \\ \Rightarrow & \{ \text{ var-left, parameter } = 7 \cdot x \ \} \\ & 5 \cdot x = -14 \\ \Rightarrow & \{ \text{ times, parameter } = \frac{1}{5} \ \} \\ & x = -14 \,/\, 5 \end{array}$$

Applying and recognizing rules

- A rule can be applied by a domain reasoner:

  $$3 \cdot (x + 4) \Rightarrow 3 \cdot x + 3 \cdot 4$$

  to give a hint or to show a worked-out solution
- and a domain reasoner can analyse a student submission to detect the application of a rule:

  $3 \cdot x + 3 \cdot 4$ OK with previous expression $3 \cdot (x + 4)$?

  Apply `distr-times` (and other rules allowed in this expression) to the previous expression and compare the result

Rules with parameters

- Some rules need extra information, such as:
  `var-left`, parameter $= 7 \cdot x$
  `times`, parameter $= \frac{1}{5}$
  ...
- This information is used to give feedback to students, for example when they ask for a hint
- It is computed within the rule
- What do we do when recognizing a different parameter?

$$12 \cdot x \equiv 7 \cdot x - 14 \Rightarrow 6 \cdot x \equiv x - 14$$

Parametrized rules

- Parametrized rules are rules that contain free meta-variables:

$$\forall\, c : a = b \;\Rightarrow\; a + c = b + c$$

  These are not proper rewrite rules anymore
- A parametrized rule can only be applied when instantiated with a particular expression for c
- We can often construct an analyser that recognizes the application of a parametrized rule in a student submission

Buggy rules

Buggy rules capture common errors of students:

$$(a+b)^2 \Rightarrow a^2+b^2$$
$$\frac{a}{b}+\frac{c}{d} \Rightarrow \frac{a+c}{b+d}$$
$$\forall\, c : a = b \Rightarrow a+c = b-c$$

- Each class of exercises has a number of buggy rules attached
- Buggy rules are tried whenever a student submission is incorrect
- Based on teachers' experience
- Can be overdone (Hennecke's thesis gives 350 buggy rules for the fraction domain)

Minor rules

Minor rules are rules without a visual effect, used to perform tasks such as:

- moving down into a term
- updating an environment, for example with the 'nice' factors found in a quadratic equation
- automatic simplification, such as replacing $x + 0$ by x
- keeping track of where we are in the solution to the exercise, for example: at form $ax^2 + bx + c = 0$, apply the quadratic formula

Minor rules play an important role when implementing the 'gestalt view' related rules that analyse the current form of the expression to determine which next step to take.

Ordering rules

- Often, many rules are applicable to an expression
- A student may apply any of the applicable rules, but which rule do we show when giving a hint?
- Which rule to show when:

$$5 \cdot (x - 1) = \frac{x - 1}{2}$$

Ordering rules

- Often, many rules are applicable to an expression
- A student may apply any of the applicable rules, but which rule do we show when giving a hint?
- Which rule to show when:

$$5 \cdot (x - 1) = \frac{x - 1}{2}$$

- We order rules to select the most relevant rule
- To solve linear equations, we could use

$$\mathit{ruleOrdering} = [\mathit{removeDivision}, \mathit{flipEquation}, \mathit{times}]$$

Outline of presentation

§4

Universiteit Utrecht

What does a step look like?

But wait:

$$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14?$$

You are doing much more in this step!

What does a step look like?

But wait:

$$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14?$$

You are doing much more in this step!

$$3 \cdot (4 \cdot x - 1) + 3$$

What does a step look like?

But wait:

$$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14?$$

You are doing much more in this step!

$$\begin{aligned} & 3 \cdot (4 \cdot x - 1) + 3 \\ \Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \end{aligned}$$

What does a step look like?

But wait:

$$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14?$$

You are doing much more in this step!

$$\begin{aligned} & 3 \cdot (4 \cdot x - 1) + 3 \\ \Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \\ \Rightarrow \quad & (12 \cdot x - 3 \cdot 1) + 3 \end{aligned}$$

What does a step look like?

But wait:

$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14$?

You are doing much more in this step!

$$
\begin{aligned}
& 3 \cdot (4 \cdot x - 1) + 3 \\
\Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3) + 3
\end{aligned}
$$

What does a step look like?

But wait:

$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14$?

You are doing much more in this step!

$$
\begin{aligned}
& 3 \cdot (4 \cdot x - 1) + 3 \\
\Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3) + 3 \\
\Rightarrow \quad & (12 \cdot x + (-3)) + 3
\end{aligned}
$$

What does a step look like?

But wait:

$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14$?

You are doing much more in this step!

$$
\begin{aligned}
& 3 \cdot (4 \cdot x - 1) + 3 \\
\Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3) + 3 \\
\Rightarrow \quad & (12 \cdot x + (-3)) + 3 \\
\Rightarrow \quad & 12 \cdot x + (-3 + 3)
\end{aligned}
$$

What does a step look like?

But wait:

$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14$?

You are doing much more in this step!

$$
\begin{aligned}
& 3 \cdot (4 \cdot x - 1) + 3 \\
\Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3) + 3 \\
\Rightarrow \quad & (12 \cdot x + (-3)) + 3 \\
\Rightarrow \quad & 12 \cdot x + (-3 + 3) \\
\Rightarrow \quad & 12 \cdot x + 0
\end{aligned}
$$

What does a step look like?

But wait:

$3 \cdot (4 \cdot x - 1) + 3 = 7 \cdot x - 14 \Rightarrow 12 \cdot x = 7 \cdot x - 14$?

You are doing much more in this step!

$$
\begin{aligned}
& 3 \cdot (4 \cdot x - 1) + 3 \\
\Rightarrow \quad & (3 \cdot 4 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3 \cdot 1) + 3 \\
\Rightarrow \quad & (12 \cdot x - 3) + 3 \\
\Rightarrow \quad & (12 \cdot x + (-3)) + 3 \\
\Rightarrow \quad & 12 \cdot x + (-3 + 3) \\
\Rightarrow \quad & 12 \cdot x + 0 \\
\Rightarrow \quad & 12 \cdot x
\end{aligned}
$$

Similar problems

- Economy of rules: I want to describe

$$a \cdot (b + c) \Rightarrow a \cdot b + a \cdot c$$

  but preferably not also:

$$a \cdot (b - c) \Rightarrow a \cdot b - a \cdot c$$
$$-a \cdot (b + c) \Rightarrow -a \cdot b - a \cdot c$$

- Canonical forms: $a + (-b)$ should be presented as $a - b$
- Granularity: users at different levels need different granularity of rules
- Recognizing user steps: when showing steps to users, we want to apply some simplifications automatically. When recognising steps, however, such simplifications are not obligatory

Views

A view views an expression in a particular format:

- a match function returns an equivalent value in a different format, for example:

$$\begin{aligned} \textit{match plusView } (a - b) &\Rightarrow a + (-b) \\ \textit{match plusView } (-(a + b)) &\Rightarrow -a + -b \end{aligned}$$

- a build function to return to the original domain, for example:

$$\begin{aligned} & 3 \cdot (4 \cdot x - 1) \\ \Rightarrow \quad & \{ \textit{ match plusView } \text{on } 4 \cdot x - 1 \ \} \\ & 3 \cdot (4 \cdot x + (-1)) \\ \Rightarrow \quad & \{ \text{ distribute } \cdot \text{ over } + \ \} \\ & 3 \cdot 4 \cdot x + 3 \cdot (-1) \\ \Rightarrow \quad & \{ \text{ simplify using } \textit{rationalView } \} \\ & 12 \cdot x - 3 \end{aligned}$$

Views and rules

- Many rules use one or more views for matching on the left-hand side
- Many rules use one or more views to clean up a result expression after rewriting
- Views and parametrized rules solve the problem of making all steps in solving an exercise explicit

Outline of presentation

§5

Universiteit Utrecht

Strategies for solving quadratic equations §5

- Naive strategy:
  Apply rewrite rules exhaustively until solved

Strategies for solving quadratic equations §5

- Naive strategy:
  Apply rewrite rules exhaustively until solved
- Algorithmic strategy:
  (1) Write the equation in the form $ax^2 + bx + c = 0$
  (2) Solve using the quadratic formula

Universiteit Utrecht

Strategies for solving quadratic equations §5

- Naive strategy:
  Apply rewrite rules exhaustively until solved
- Algorithmic strategy:
  (1) Write the equation in the form $ax^2 + bx + c = 0$
  (2) Solve using the quadratic formula
- Expert ('Gestalt view') strategy:
  (1) Inspect the form of the equation
  (2) Possibly rewrite the equation into a nice form
  (3) Based on the form of the equation, use the appropriate substrategy:
    (3a) In case no linear term: take square root
    (3b) In case no constant term: factorise and solve
    (3c) In case nice factors: factorise and solve
    (3d) Use quadratic formula

A strategy specification language §5

We need the following concepts for specifying a strategy:

- apply a basic rewrite rule, *("· distributes over +")*
- sequence *("first . . . then . . . ")* $s \mathbin{<\star>} t$
- choice *("use one of the substrategies for ...")* $s \mathbin{<|>} t$
- unit elements *succeed*, *fail*
- labels *label* ℓ s
- recursion *fix* f
- apply exhaustively *("repeat . . . as long as possible")* *repeat*
- traversals *("apply . . . top down")* *topDown*

Universiteit Utrecht

A strategy for solving quadratic equations §5

```
quadraticStrategyG =
   label "Quadratic Equation Strategy" $ repeatS $
      -- Relaxed strategy: even if there are "nice" factors,
      -- allow use of quadratic formula
      somewhere (generalForm <|> generalABCForm)
       |> somewhere zeroForm
       |> somewhere constantForm
       |> simplifyForm
       |> topForm
   where
         -- ax^2 + bx + c == 0, without quadratic formula
      generalForm = label "general form" ...

      generalABCForm = ...

      ...
```

Universiteit Utrecht

Domain reasoners: the main idea

- Use rewrite strategies for exercises
- A strategy describes valid sequences of rules
- View a strategy specification as a context-free grammar
- Track intermediate steps by means of parsing

Domain reasoners: the main idea

- Use rewrite strategies for exercises
- A strategy describes valid sequences of rules
- View a strategy specification as a context-free grammar
- Track intermediate steps by means of parsing

Feedback service	Parsing problem
finished	is the empty sentence (ϵ) accepted?
provide hint	compute the "first set"
worked-out solution	construct a sentence
diagnosis	try to recognize the rewrite rule that was used, and parse this rule as the next symbol of the input

No new thing under the sun?

§5

Many approaches to plan recognition transform it into a parsing problem. A grammar specifies how plans are decomposed into actions and sub-actions, and a particular sequence of observations is regarded as a sentence to be parsed with respect to this grammar.

From: Formal Approaches to Student Modelling, John Self, 1994.

Universiteit Utrecht

Contributions

A strategy language should be generic, support the automatic calculation of feedback, satisfy the cognitive fidelity principle, and be observable, adaptable, and composable.

Our contribution is:

- Observability of strategies, which makes it possible to
  - communicate a strategy
  - verify a strategy
  - adapt a strategy
- Compositionality of strategies, which makes it easier to combine and reuse strategies

Outline of presentation

Computer algebra systems

Computer algebra systems (CASs) are almost useless for interactive exercises.

- A CAS can calculate a final answer, and use that to check correctness of a step
- A CAS has no clue about intermediate steps
- It is hard to determine if an expression is a solution: *factor* $(x^2 - 6\,x + 9)$ gives $(x - 3)^2$, but what about

$$(3 - x)^2$$
$$(x - 3)\,(x - 3)$$
$$(3 - x)\,(3 - x)$$
$$9 \cdot (1 - \tfrac{x}{3})^2$$

- A domain reasoner can be viewed as an idiosyncratic CAS

Specifying feedback per exercise §6

Specifying feedback per exercise is infeasible:

- authoring size: each exercise has to be extended with all feedback possibilities: can easily blow up every exercise by a factor of 10 to 20. Generation might help here.
- maintenance: strategies are regularly extended or adapted. If strategies and feedback are encoded in exercises you get a maintenance nightmare
- adaptivity: 'my students can solve a linear equation in a single step'...
- student model: there needs to be central control on feedback to be able to report on student progress

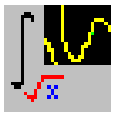
Other domain reasoners

Carnegie learning: Algebra Cognitive tutor

- Advanced, commercial tool, widely used in the US

Superconnie

- Advanced set of tools for editing maths and interactive exercises



Mathpert

- Applying rewrite rules to (sub)expressions. No free editing

None of these use observable, compositional rewrite strategies.

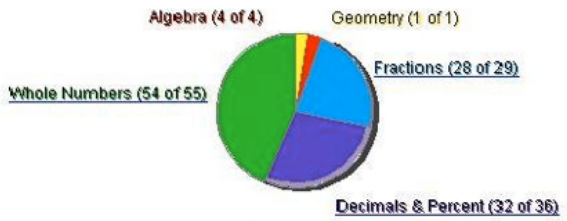
Outline of presentation

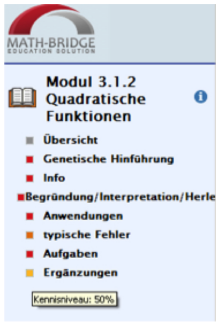
§7

Universiteit Utrecht

Future work: student model

- Existing tools with student models:





- Student models are rather coarse, and say nothing about particular misconceptions
- Our domain reasoner gets very detailed information about student interactions: develop a student model based on these interactions
- Use this student model to make students work on particular misconceptions

Future work: automatic assessment

- Automatic assessment is applied standard in multiple choice questions, and sometimes also based on final answers using a CAS.
- On paper, teachers correct derivations
- We have performed automatic assessment in the programming domain. True positives: 89%
- For mathematics this is promising too: filter out correct solutions and solutions with common errors to focus on real problems

Future work: adapting domain reasoners

- Students are different

Future work: adapting domain reasoners §7

- Students are different
- Teachers are different

Universiteit Utrecht

Future work: adapting domain reasoners

- Students are different
- Teachers are different
- How can we best support adaptation of our strategies, both based on student model, and based on teacher requirements?

Conclusions

- We have developed a framework based on rewrite rules, views, and strategies, which can be used to relatively easily develop domain reasoners
- Our domain reasoners are used by quite a few learning environments for mathematics, such as the Freudenthal Institute's DWO
- By making algebraic procedural knowledge explicit in open-source software, we think we can make big steps forward in helping a student learn algebra

More info

- `http://ideas.cs.uu.nl/`
- J.T.Jeuring@uu.nl

Universiteit Utrecht