Automatic Generation and Delivery of Multiple-Choice Math Quizzes

Ana Paula Tomás [1] José Paulo Leal [2]

[1] DCC-FCUP & CMUP

[2] DCC-FCUP & CRACS-INESC TEC

U.PORTO

CP 2013, Uppsala, Sweden

Outline

Struggling with Mathematics

$|x| > 6 \Leftrightarrow x < -6 \vee x > 6?$
$|x| > 6 \Leftrightarrow x < -6 \wedge x > 6?$

Easy! Rotate $>$ 90° clockwise.

Strive for understanding as opposed to memorization.

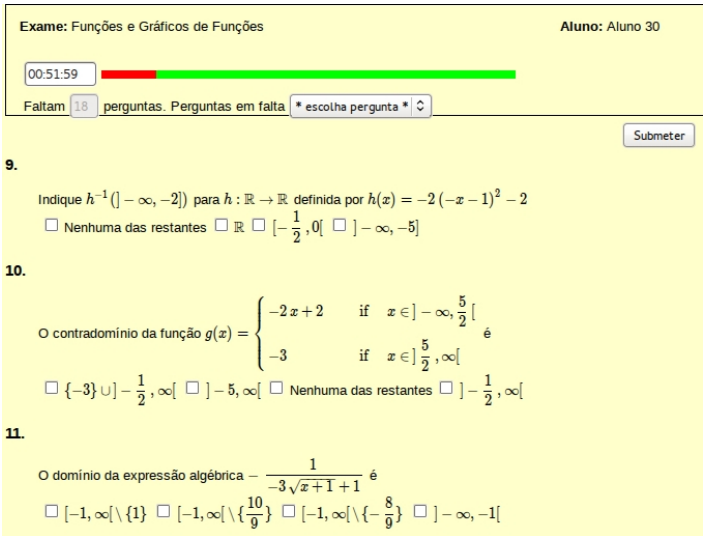
- Computer aided training and assessment for facing insufficient competencies in mathematics of school leavers and students enrolled in initial training within Sci/Tech University curricula [Math-Bridge, e-Learning platform]
- To create drills by hand is time consuming and error prone. Develop tools to create and solve drills automatically.
- There exist hundreds of exercise assistants online.
- Avoid banks of repetitive exercises where only a few parameters are different.

A Remedial Pre-Calculus Course at FCUP

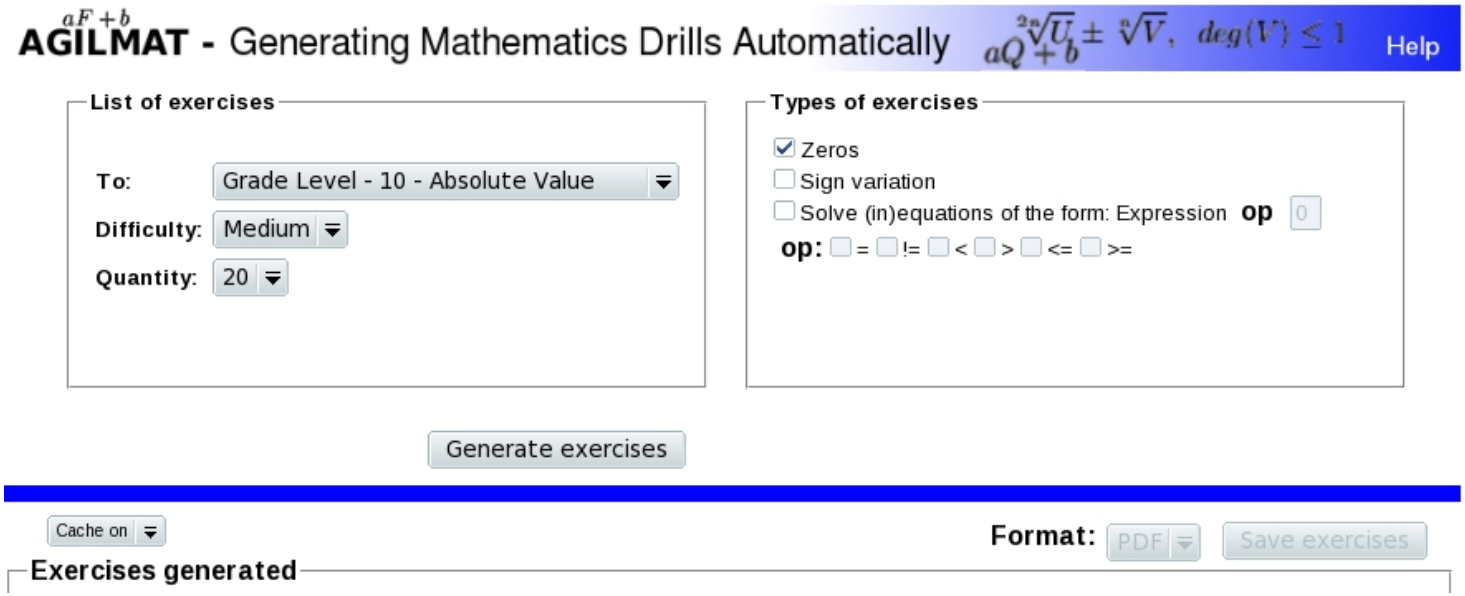
- A four week course (4h/week) designed to help first year students of different courses address gaps in basic topics in algebra and trigonometry.
- Strongly recommended for all students who have a low score on a diagnostic test.
- Online multiple-choice quizzes carried out every week for formative assessment, in-class and out-of-class.

Syllabus

Real numbers. Basic algebraic operations: fractions, exponents and radicals. Linear, quadratic and rational equations and inequations. Elementary functions. Composite and inverse. Polynomials. Trigonometric functions and identities. Exponential and logarithmic functions. Sequences and limits. Limits of functions.

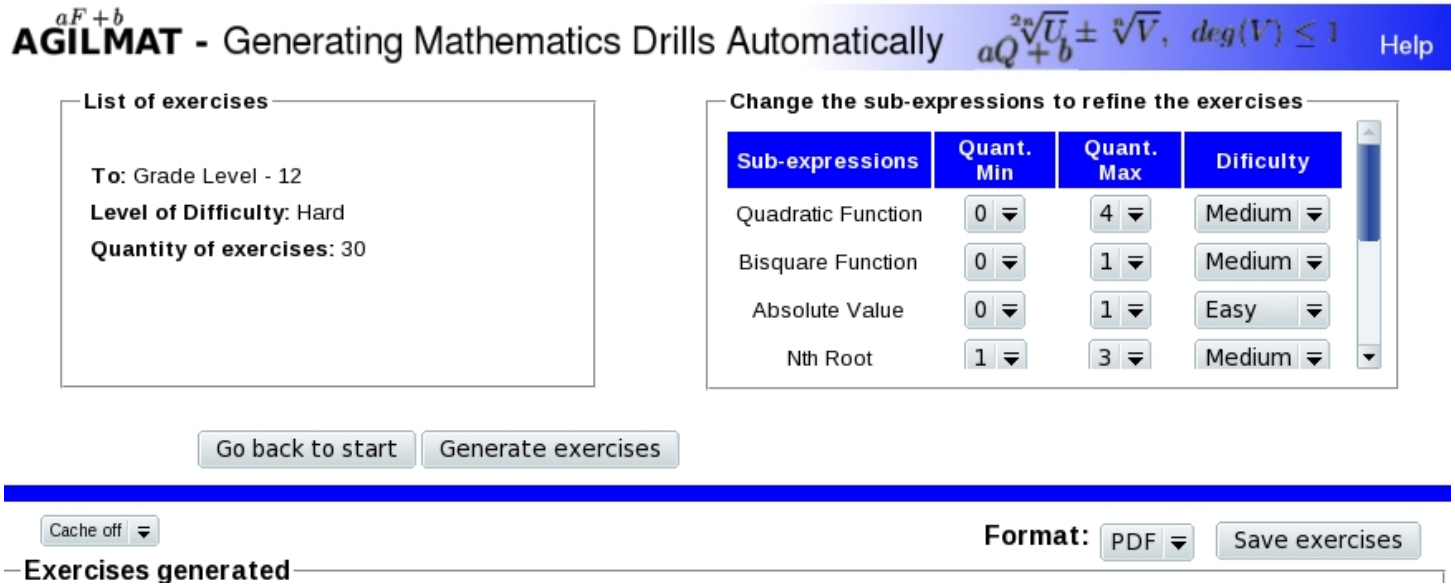


AGILMAT Prototype (www.dcc.fc.up.pt:8080/Agilmat)



- Available through a wizard-based configuration interface, with progressive disclosure of (choice-dependent) parameters
- Supports distinct initial settings – profiles – by parametrizing the interface, generator and solver, through XML configuration files.

AGILMAT Prototype (www.dcc.fc.up.pt:8080/Agilmat)



- The selected default profile specifies the parameters, their range and default values. Users may refine or redefine parameters.
- AGILMAT may potentially create hundreds of instances and their one-line solutions. The kernel is an application of CLP.

AGILMAT Prototype (www.dcc.fc.up.pt:8080/Agilmat)

A basic design principle

We do not need to support full generality to obtain an useful tool.

FIND THE DOMAIN OF THE EXPRESSION

$$-2\sqrt{\frac{x^2-3x-1}{-3x-3}}-3$$

SOLUTION:

$$]-\infty,-1[\cup]\frac{3}{2}-\frac{1}{2}\sqrt{13},\frac{3}{2}+\frac{1}{2}\sqrt{13}[$$

FIND THE RANGE OF

$$r(x)=(-3x-2)\ (2x+1)$$

SOLUTION:

$$]-\infty,\frac{1}{24}]$$

FIND THE SET OF SOLUTIONS OF

$$\frac{x^2-2x-2}{|\sqrt{-3x+1}|}<0$$

SOLUTION:

$$]1-\sqrt{3},\frac{1}{3}[$$

FIND THE SET OF SOLUTIONS OF

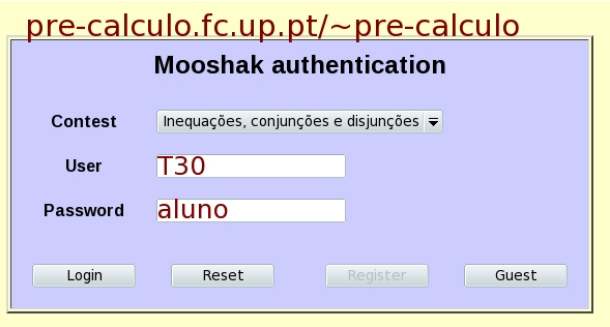
$$|-x^6-2x^3-1|\neq 2$$

SOLUTION:

$$]-\infty,\infty[\setminus\{-\sqrt[3]{1+\sqrt{2}},\sqrt[3]{-1+\sqrt{2}}\}$$

The **expression generator** follows a grammar and supports constraints to tailor the expressions to special needs. The **exercise generator and solver** implements procedures students learn to find exact solutions.

AgilmatQuiz



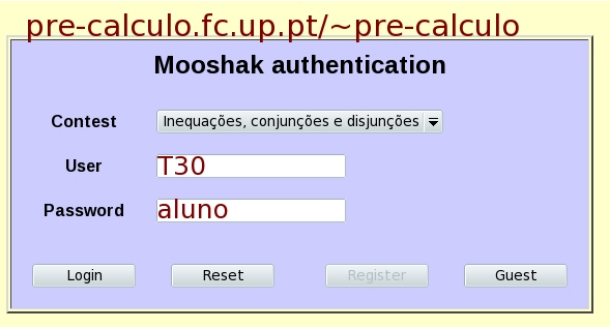
Mooshak learning system used for quiz delivery. A quiz consists of groups of questions created by hand and by AgilmatQuiz, in a LaTeX format (`mooshakquiz` style).

```
\begin{quizgroup}{1}
\begin{quizquestion} The set of solutions of  $-2\,x^{2}+x \leq 1$ is
\begin{quizchoice}{false} $\{\}$  \end{quizchoice}
\begin{quizchoice}{false} $ ]-\infty,0] \,\cup \,[\frac{1}{2},\infty[ $ \end{quizchoice}
\begin{quizchoice}{true} $\mathbb{R}$  \end{quizchoice}
\begin{quizchoice}{false} None of the above \end{quizchoice}
\end{quizquestion} ... \begin{quizquestion}...\end{quizquestion} \end{quizgroup}
```

Adapt AGILMAT kernel to create new exercises, produce plausible wrong answers, and select the wrong answers to each quiz question.

Topics covered so far: **algebraic functions taught at high-school**, the notions of domain, range, reciprocal image, piecewise-defined functions and the solution of equations and inequalities.

AgilmatQuiz



Mooshak learning system used for quiz delivery. A quiz consists of groups of questions created by hand and by AgilmatQuiz, in a LaTeX format (`mooshakquiz` style).

```
\begin{quizgroup}{1}
\begin{quizquestion} The set of solutions of  $-2\,x^{2}+x \leq 1$ is
\begin{quizchoice}{false} $\{\}$  \end{quizchoice}
\begin{quizchoice}{false} $ ]-\infty,0] \,\cup \,[\frac{1}{2},\infty[ $ \end{quizchoice}
\begin{quizchoice}{true} $\mathbb{R}$  \end{quizchoice}
\begin{quizchoice}{false} None of the above \end{quizchoice}
\end{quizquestion} ... \begin{quizquestion}...\end{quizquestion} \end{quizgroup}
```

Adapt AGILMAT kernel to create new exercises, produce plausible wrong answers, and select the wrong answers to each quiz question.

Topics covered so far: **algebraic functions taught at high-school**, the notions of domain, range, reciprocal image, piecewise-defined functions and the solution of equations and inequalities.

AGILMAT Kernel (Generators and Solver)

Expression Generator:

calls `examples(File,DegI,RateMin-RateMax,X,NumbI)` to yield a `File` of expressions in the variable `X`, of degree `DegI` and difficulty level within `RateMin-RateMax`, with `NumbI` instances of each type (*pattern*).

Exercise Generator & Solver:

uses this `File` to create a sheet of exercises with their one-line solutions. To handle some nonlinear constraints and compute exact solutions, performs symbolic processing. A set of tools (modules) fairly less complex than a computer algebra system.

From Patterns to Expressions

prodexpr	$\rightarrow$	*factor* | *factor* $*$ *prodexpr*
factor	$\rightarrow$	*sumexpr* | *basic*
basic	$\rightarrow$	$ipol_2$(x) | *bisqr*
	$\rightarrow$	*fbasic* | $fpol_1$(*fbasic*) | $fpol_1$(x)
fbasic	$\rightarrow$	abs(*basic*) | pow(*basic*, *N*)
	$\rightarrow$	rad(*basic*, *N*)
$ipol_2(T)$	$\rightarrow$	pol(T, [a, b, c]), $abc \neq 0$
$fpol_1(T)$	$\rightarrow$	pol(T, [a, b]), $a \neq 0$
bisqr	$\rightarrow$	$ipol_2$(pow(x, *N*)), $N \geq 2$

```
% T is a type
prodstype(T,G,Rate,CTs,Ops) :-
  constrs(CTs,urestrs_factor),
  factorstype(T,G,Rate,CTs,Ops).
prodstype(Tb*T,G,Rate,CTs,Ops) :-
  rate_restr(prodstype,Rate,[RateB,RateT]),
  types_restr(prodstype,CTs,[CTsB,CTsT]),
  ops_restr(Ops,[1,OpsB,OpsT]), OpsT #=< OpsB,
  Gb #>= 1, Gt #>= 1, G #= Gt+Gb,
  constrs(CTsB,urestrs_factor),
  factorstype(Tb,Gb,RateB,CTsB,OpsB),
  prodstype(T,Gt,RateT,CTsT,OpsT).
```

Idea: $\sqrt[N]{(ax+b)^M}$ is rad(pow(pol(x, [a, b]), M), N) and corresponds to **type/pattern** rad(N) o pow(M) o p1 o x.

```
rad(3)o(abs o p1 o pow(2)o p1 o x + pow(2)o p1 o x)
rad(abs(pol(pow(pol(x,[3,-4]),2),[-4,-2])) + pow(pol(x,[1,-3]),2),3)
```

$$\sqrt[3]{|-4(3x-4)^2-2|+(x-3)^2}$$

Create a type and then instances of expressions of that type.

From Patterns to Expressions

prodexpr	$\rightarrow$	*factor* | *factor* $*$ *prodexpr*
factor	$\rightarrow$	*sumexpr* | *basic*
basic	$\rightarrow$	$ipol_2$(x) | *bisqr*
	$\rightarrow$	*fbasic* | $fpol_1$(*fbasic*) | $fpol_1$(x)
fbasic	$\rightarrow$	abs(*basic*) | pow(*basic*, *N*)
	$\rightarrow$	rad(*basic*, *N*)
$ipol_2(T)$	$\rightarrow$	pol(T, [a, b, c]), $abc \neq 0$
$fpol_1(T)$	$\rightarrow$	pol(T, [a, b]), $a \neq 0$
bisqr	$\rightarrow$	$ipol_2$(pow(x, *N*)), $N \geq 2$

```
% T is a type
prodstype(T,G,Rate,CTs,Ops) :-
  constrs(CTs,urestrs_factor),
  factorstype(T,G,Rate,CTs,Ops).
prodstype(Tb*T,G,Rate,CTs,Ops) :-
  rate_restr(prodstype,Rate,[RateB,RateT]),
  types_restr(prodstype,CTs,[CTsB,CTsT]),
  ops_restr(Ops,[1,OpsB,OpsT]), OpsT #=< OpsB,
  Gb #>= 1, Gt #>= 1, G #= Gt+Gb,
  constrs(CTsB,urestrs_factor),
  factorstype(Tb,Gb,RateB,CTsB,OpsB),
  prodstype(T,Gt,RateT,CTsT,OpsT).
```

Idea: $\sqrt[N]{(ax+b)^M}$ is rad(pow(pol(x, [a, b]), M), N) and corresponds to **type/pattern** rad(N) o pow(M) o p1 o x.

```
rad(3)o(abs o p1 o pow(2)o p1 o x + pow(2)o p1 o x)
rad(abs(pol(pow(pol(x,[3,-4]),2),[-4,-2])) + pow(pol(x,[1,-3]),2),3)
```

$$\sqrt[3]{|-4(3x-4)^2-2|+(x-3)^2}$$

Create a type and then instances of expressions of that type.

Configuration Through Constraints on FD Variables

```
prodstype(T,G,Rate,CTs,Ops) :-
  constrs(CTs,urestrs_factor),
  factorstype(T,G,Rate,CTs,Ops).
prodstype(Tb*T,G,Rate,CTs,Ops) :-
  rate_restr(prodstype,Rate,[RateB,RateT]),
  types_restr(prodstype,CTs,[CTsB,CTsT]),
  ops_restr(Ops,[1,OpsB,OpsT]), OpsT #=< OpsB,
  Gb #>= 1, Gt #>= 1, G #= Gt+Gb,
  constrs(CTsB,urestrs_factor),
  factorstype(Tb,Gb,RateB,CTsB,OpsB),
  prodstype(T,Gt,RateT,CTsT,OpsT).
```

```
constrs(CTs,Functor) :-
  Goal =.. [Functor,CTsCtr],
  call(Goal),
  single_vars_low_up_constrs(CTsCtr,CTs),
  user_other_restrs(Functor,CTs).

rate_restr(T,Rate,L) :-
  nonnegative(L),
  user_rate(T,Rt),
  sum([Rt|L],#=,Rate).
```

The current configuration constraints

Simple value or arithmetic constraints, lower and upper bound cardinality constraints or conditional ones, e.g., $l \leq x_i \leq u$, $l \leq \sum_{i \in I_1} x_i \leq u$, and $\sum_{i \in I_1} x_i \geq l \Rightarrow \sum_{i \in I_2} x_i \leq u$, with $x_i \in \mathbb{Z}_0^+$.

A dynamic CSP model: New variables and constraints are added during the execution.

New Forms of Expressions and Exercises

Piecewise functions: control branch connections. First fix the coefficients and then the breakpoints and domains.

```
piecewise([abs o p1 o x,p1 o x]).
piecewise([br(abs(pol(x,[-2,-1])),_),br(pol(x,[-5,-4]),_)]).
```

$$\begin{cases} |-2x-1| & \text{if } x \in ? \\ -5x-4 & \text{if } x \in ? \end{cases}$$

```
piecewise([abs o p1 o x,p1 o x]).
piecewise([br(abs(pol(x,[-2,-1])),[a(-(infty)),a(rat(-1,1))]),
           br(pol(x,[-5,-4]),[f(rat(-1,1)),a(infty)])]).
```

$$\begin{cases} |-2x-1| & \text{if } x \in]-\infty,-1[\\ -5x-4 & \text{if } x \in [-1,\infty[\end{cases}$$

New exercises: more expressions/dependencies.

Find the solution set of $3x-4 \leq 0 \vee -2x-7 \geq 0$.
Find $h^{-1}(]-\infty,-2])$ for $h : \mathbb{R} \to \mathbb{R}$ defined by $h(x) = -2(-x-1)^2 - 2$.

```
groups(p(quizConjDisj,2,1)-p1).
groups(p(quizDomains,1)-rad1).
% groups(p(Pred,Args,Nq)-File)

quiz(FileOut) :-
  tell(FileOut),
  openenv(mooshakquiz),
  findall(G,groups(G),QuizGs),
  quiz_agm(QuizGs),
  closeenv(mooshakquiz), told.

quiz_agm([]).
quiz_agm([QuizG-File|QuizGs]) :-
  see(File),
  quizgroup_agilmat(QuizG),
  seen, quiz_agm(QuizGsR).
```

```
quizDomains(T,E,X) :-
  Set = [a(-infty),a(infty)],
  ws_dom(E,X,Set,S,WSols), !,
  select(WSols,S,false-'None',Chs),
  Chs = [_A,_B,_C,_D],
  writeQuestion(dom(T,E,X),Chs).
```

New Forms of Expressions and Exercises

Piecewise functions: control branch connections. First fix the coefficients and then the breakpoints and domains.

```
piecewise([abs o p1 o x,p1 o x]).
piecewise([br(abs(pol(x,[-2,-1])),_),br(pol(x,[-5,-4]),_)]).
```

$$\begin{cases} |-2x-1| & \text{if } x \in ? \\ -5x-4 & \text{if } x \in ? \end{cases}$$

```
piecewise([abs o p1 o x,p1 o x]).
piecewise([br(abs(pol(x,[-2,-1])),[a(-(infty)),a(rat(-1,1))]),
           br(pol(x,[-5,-4]),[f(rat(-1,1)),a(infty)])]).
```

$$\begin{cases} |-2x-1| & \text{if } x \in]-\infty,-1[\\ -5x-4 & \text{if } x \in [-1,\infty[\end{cases}$$

New exercises: more expressions/dependencies.

Find the solution set of $3x-4 \leq 0 \vee -2x-7 \geq 0$.
Find $h^{-1}(]-\infty,-2])$ for $h : \mathbb{R} \to \mathbb{R}$ defined by $h(x) = -2(-x-1)^2 - 2$.

```
groups(p(quizConjDisj,2,1)-p1).
groups(p(quizDomains,1)-rad1).
% groups(p(Pred,Args,Nq)-File)

quiz(FileOut) :-
  tell(FileOut),
  openenv(mooshakquiz),
  findall(G,groups(G),QuizGs),
  quiz_agm(QuizGs),
  closeenv(mooshakquiz), told.

quiz_agm([]).
quiz_agm([QuizG-File|QuizGs]) :-
  see(File),
  quizgroup_agilmat(QuizG),
  seen, quiz_agm(QuizGsR).
```

```
quizDomains(T,E,X) :-
  Set = [a(-infty),a(infty)],
  ws_dom(E,X,Set,S,WSols), !,
  select(WSols,S,false-'None',Chs),
  Chs = [_A,_B,_C,_D],
  writeQuestion(dom(T,E,X),Chs).
```

New Forms of Expressions and Exercises

Piecewise functions: control branch connections. First fix the coefficients and then the breakpoints and domains.

```
piecewise([abs o p1 o x,p1 o x]).
piecewise([br(abs(pol(x,[-2,-1])),_),br(pol(x,[-5,-4]),_)]).
```

$$\begin{cases} |-2x-1| & \text{if } x \in ? \\ -5x-4 & \text{if } x \in ? \end{cases}$$

```
piecewise([abs o p1 o x,p1 o x]).
piecewise([br(abs(pol(x,[-2,-1])),[a(-(infty)),a(rat(-1,1))]),
           br(pol(x,[-5,-4]),[f(rat(-1,1)),a(infty)])]).
```

$$\begin{cases} |-2x-1| & \text{if } x \in]-\infty,-1[\\ -5x-4 & \text{if } x \in [-1,\infty[\end{cases}$$

New exercises: more expressions/dependencies.

Find the solution set of $3x-4 \leq 0 \vee -2x-7 \geq 0$.
Find $h^{-1}(]-\infty,-2])$ for $h : \mathbb{R} \to \mathbb{R}$ defined by $h(x) = -2(-x-1)^2 - 2$.

```
groups(p(quizConjDisj,2,1)-p1).
groups(p(quizDomains,1)-rad1).
% groups(p(Pred,Args,Nq)-File)

quiz(FileOut) :-
  tell(FileOut),
  openenv(mooshakquiz),
  findall(G,groups(G),QuizGs),
  quiz_agm(QuizGs),
  closeenv(mooshakquiz), told.

quiz_agm([]).
quiz_agm([QuizG-File|QuizGs]) :-
  see(File),
  quizgroup_agilmat(QuizG),
  seen, quiz_agm(QuizGsR).
```

```
quizDomains(T,E,X) :-
  Set = [a(-infty),a(infty)],
  ws_dom(E,X,Set,S,WSols), !,
  select(WSols,S,false-'None',Chs),
  Chs = [_A,_B,_C,_D],
  writeQuestion(dom(T,E,X),Chs).
```

Plausible Wrong Answers (distractors)

Incorporate common errors of students in distractors.
Adapt AGILMAT symbolic solver and some of its submodules to include buggy rules that translate such errors. Add variables to restrict and track the number of wrong rules applied.

```
% correct version
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :-
 !, domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :-
 !, (even(N) ->
  (domain_expr(U,X,Dom,DomU),
   solve(DomU,U,geq,rat(0,1),X,Domf));
  domain_expr(U,X,Dom,Domf)).
```

The domain of ...

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = x$ is $\mathbb{R}$.

$f : \mathbb{R}_0^+ \to \mathbb{R}$
s.t. $f(x) = x$ is $\mathbb{R}_0^+$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5x + 3$ is $\mathbb{R}$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5\sqrt{x} + 3$ is $\mathbb{R}_0^+$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5\sqrt[3]{x} + 3$ is $\mathbb{R}$.

Plausible Wrong Answers (distractors)

Incorporate common errors of students in distractors.
Adapt AGILMAT symbolic solver and some of its submodules to include buggy rules that translate such errors. Add variables to restrict and track the number of wrong rules applied.

```
% correct version
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :-
 !, domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :-
 !, (even(N) ->
  (domain_expr(U,X,Dom,DomU),
   solve(DomU,U,geq,rat(0,1),X,Domf));
  domain_expr(U,X,Dom,Domf)).
```

The domain of ...

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = x$ is $\mathbb{R}$.

$f : \mathbb{R}_0^+ \to \mathbb{R}$
s.t. $f(x) = x$ is $\mathbb{R}_0^+$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5x + 3$ is $\mathbb{R}$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5\sqrt{x} + 3$ is $\mathbb{R}_0^+$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5\sqrt[3]{x} + 3$ is $\mathbb{R}$.

Plausible Wrong Answers (distractors)

Incorporate common errors of students in distractors.
Adapt AGILMAT symbolic solver and some of its submodules to include buggy rules that translate such errors. Add variables to restrict and track the number of wrong rules applied.

```
% correct version
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :-
 !, domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :-
 !, (even(N) ->
  (domain_expr(U,X,Dom,DomU),
   solve(DomU,U,geq,rat(0,1),X,Domf));
  domain_expr(U,X,Dom,Domf)).
```

The domain of ...

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = x$ is $\mathbb{R}$.

$f : \mathbb{R}_0^+ \to \mathbb{R}$
s.t. $f(x) = x$ is $\mathbb{R}_0^+$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5x + 3$ is $\mathbb{R}$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5\sqrt{x} + 3$ is $\mathbb{R}_0^+$.

$f : \mathbb{R} \to \mathbb{R}$
s.t. $f(x) = 5\sqrt[3]{x} + 3$ is $\mathbb{R}$.

Plausible Wrong Answers (distractors)

Incorporate common errors of students in distractors.
Adapt AGILMAT symbolic solver and some of its submodules to include buggy rules that translate such errors. Add variables to restrict and track the number of wrong rules applied.

```
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :- !,
  domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :- !,
  (even(N) ->
     (domain_expr(U,X,Dom,DomU),
       solve(DomU,U,geq,rat(0,1),X,Domf));
      domain_expr(U,X,Dom,Domf)).
```

E.g., The domain of $f(x) = -3\sqrt[3]{-x+2} - 3$ is

☐ $\mathbb{R}$ ☐ $\{2\}$ ☐ $]-\infty, 2]$ ☐ $\{\}$

```
% reals = [a(-infty),a(infty)]
wrg_domain_expr(X,X,Dom,Dom,W) :- {W = 0}.
wrg_domain_expr(X,X,Dom,reals,Wf) :- !,
   {Wf = 1},  Dom \= reals.
wrg_domain_expr(pol(U,_L),X,Dom,Domf,Wf) :-
   {Wf >= 0}, wrg_domain_expr(U,X,Dom,Domf,Wf).
wrg_domain_expr(pol(U,_L),X,Dom,Domf,Wf) :- !,
   {Wf = 1, Ok = 0},
   wrg_domain_expr(U,X,Dom,DomS,Ok),
   DomS \= reals, Domf = reals.
wrg_domain_expr(rad(U,_N),X,Dom,Domf,Wf) :-
   odd(N), !, {W >= 0, Wi >=0, Wf = W+Wi},
   wrg_domain_expr(U,X,Dom,Domi,W),
   ((({Wi=0}, Domf = Domi);
     ({Wi=1}, Domf = Dom, Domi \= Dom);
     ({Wi=1+Wii,Wii>=0},
       wrg_solve(Domi,U,geq,rat(0,1),X,Domf,Wii))).
...
```

Plausible Wrong Answers (distractors)

Incorporate common errors of students in distractors.
Adapt AGILMAT symbolic solver and some of its submodules to include buggy rules that translate such errors. Add variables to restrict and track the number of wrong rules applied.

```
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :- !,
  domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :-  !,
  (even(N) ->
     (domain_expr(U,X,Dom,DomU),
        solve(DomU,U,geq,rat(0,1),X,Domf));
      domain_expr(U,X,Dom,Domf)).
```

E.g., The domain of
$f(x) = -3\sqrt[3]{-x+2} - 3$ is
$\square\ \mathbb{R}\quad \square\ \{2\}\quad \square\ ]-\infty, 2]\quad \square\ \{\}$

```
% reals = [a(-infty),a(infty)]
wrg_domain_expr(X,X,Dom,Dom,W) :- {W = 0}.
wrg_domain_expr(X,X,Dom,reals,Wf) :- !,
   {Wf = 1},  Dom \= reals.
wrg_domain_expr(pol(U,_L),X,Dom,Domf,Wf) :-
   {Wf >= 0}, wrg_domain_expr(U,X,Dom,Domf,Wf).
wrg_domain_expr(pol(U,_L),X,Dom,Domf,Wf) :- !,
   {Wf = 1, Ok = 0},
   wrg_domain_expr(U,X,Dom,DomS,Ok),
   DomS \= reals, Domf = reals.
wrg_domain_expr(rad(U,_N),X,Dom,Domf,Wf) :-
   odd(N), !, {W >= 0, Wi >=0, Wf = W+Wi},
   wrg_domain_expr(U,X,Dom,Domi,W),
   ((({Wi=0}, Domf = Domi);
     ({Wi=1}, Domf = Dom, Domi \= Dom);
     ({Wi=1+Wii,Wii>=0},
       wrg_solve(Domi,U,geq,rat(0,1),X,Domf,Wii))).
...
```

Plausible Wrong Answers (distractors)

Incorporate common errors of students in distractors.
Adapt AGILMAT symbolic solver and some of its submodules to include buggy rules that translate such errors. Add variables to restrict and track the number of wrong rules applied.

```
domain_expr(X,X,Dom,Dom) :- !.
domain_expr(pol(U,_L),X,Dom,Domf) :- !,
  domain_expr(U,X,Dom,Domf).
domain_expr(rad(U,N),X,Dom,Domf) :-  !,
  (even(N) ->
     (domain_expr(U,X,Dom,DomU),
        solve(DomU,U,geq,rat(0,1),X,Domf));
      domain_expr(U,X,Dom,Domf)).
```

E.g., The domain of
$f(x) = -3\sqrt[3]{-x+2} - 3$ is
$\square\ \mathbb{R}\quad \square\ \{2\}\quad \square\ ]-\infty, 2]\quad \square\ \{\}$

```
% reals = [a(-infty),a(infty)]
wrg_domain_expr(X,X,Dom,Dom,W) :- {W = 0}.
wrg_domain_expr(X,X,Dom,reals,Wf) :- !,
   {Wf = 1},  Dom \= reals.
wrg_domain_expr(pol(U,_L),X,Dom,Domf,Wf) :-
   {Wf >= 0}, wrg_domain_expr(U,X,Dom,Domf,Wf).
wrg_domain_expr(pol(U,_L),X,Dom,Domf,Wf) :- !,
   {Wf = 1, Ok = 0},
   wrg_domain_expr(U,X,Dom,DomS,Ok),
   DomS \= reals, Domf = reals.
wrg_domain_expr(rad(U,_N),X,Dom,Domf,Wf) :-
   odd(N), !, {W >= 0, Wi >=0, Wf = W+Wi},
   wrg_domain_expr(U,X,Dom,Domi,W),
   ((({Wi=0}, Domf = Domi);
     ({Wi=1}, Domf = Dom, Domi \= Dom);
     ({Wi=1+Wii,Wii>=0},
       wrg_solve(Domi,U,geq,rat(0,1),X,Domf,Wii))).
...
```

Selecting Distractors

- An heuristic rather than a preference model; random choice, biased by the number of derivations leading to distractor, its intersection with correct answer, preference for some rule,...
- Preference for distractors resulting from a single error. Discard options that can give clues for looking more absurd.
- Find all wrong solutions by backtracking. Filter them afterwards to discard repetitions (and the ones equal to the correct one).
- The exact comparison of solutions can be undecidable. The solver is not complete as a domain reasoner although it handles restricted forms. Alternative derivations may lead to equivalent expressions, both canonical ($2\sqrt{7+4\sqrt{3}}$ and $4+2\sqrt{3}$).
- "None of the above" can occur either as a distractor or correct answer, making the guess by a simple analysis of the offered answers more difficult.

Conclusions

- An application where the use of declarative languages was crucial for a rapid development of an useful tool. CP makes easier the re-usability and customization of the system.
- About 7000 lines of code for the generator and solver, at the current stage (proof of concept).
- Address a specific type of multiple-choice questions. Extend the approach to other quiz types or to populate question banks.

Future Work

- Consider other topics (e.g., transcendental functions).
- Study execution models where constraint propagation has more impact in program transformation.
- Devise other strategies for selection of distractors.
- Evaluate the application (e.g., can teachers separate AgilmatQuiz questions from ones produced manually?).

Thank you for your attention!

Future Work

- Consider other topics (e.g., transcendental functions).
- Study execution models where constraint propagation has more impact in program transformation.
- Devise other strategies for selection of distractors.
- Evaluate the application (e.g., can teachers separate AgilmatQuiz questions from ones produced manually?).

Thank you for your attention!

Some Links

- The AGILMAT Prototype:

  `http://www.dcc.fc.up.pt:8080/Agilmat`

- Examples of exercises created by AgilmatQuiz:

  `http://www.dcc.fc.up.pt/~apt/Research/AgilmatQuiz.html`

- Multiple-choice quizzes for the Pre-Calculus course (in Portuguese)

  `http://pre-calculo.fc.up.pt/~pre-calculo`

  User: T30 Password: aluno

  All exercises of the contests “Inequações, conjunções e disjunções” and “Funções e Gráficos de funções” were generated by AgilmatQuiz, except the ones about function graphs that were created by colleagues of the Department of Mathematics.

- The Mooshak system

  `http://mooshak.dcc.fc.up.pt/`