Engineering Mathematics – – intuitive and formal

Walther Neuper

IICM, Institute for Computer Media,
University of Technology.
Graz, Austria

eduTPS: JUSTIFYING MATH –
Working Group on Education and TP Technology
at CADGME, Targu Mures, Romania
September 7, 2016

Outline

Introduction to $\mathcal{ISAC}$

- $\mathcal{ISAC}$ is designed for “pure” mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” in Isabelle
  3. propose a next step if learners get stuck: “next-step-guidance” due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to *ISAC*

- *ISAC* is designed for "pure" mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the "LCF-paradigm" in Isabelle
  3. propose a next step if learners get stuck: "next-step-guidance" due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to $\mathcal{ISAC}$

- $\mathcal{ISAC}$ is designed for "pure" mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the "LCF-paradigm" in Isabelle
  3. propose a next step if learners get stuck: "next-step-guidance" due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to $\mathcal{ISAC}$

- $\mathcal{ISAC}$ is designed for "pure" mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the "LCF-paradigm" in Isabelle
  3. propose a next step if learners get stuck: "next-step-guidance" due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to *ISAC*

- *ISAC* is designed for “pure” mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” in Isabelle
  3. propose a next step if learners get stuck: “next-step-guidance” due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to *ISAC*

- *ISAC* is designed for "pure" mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the "LCF-paradigm" in Isabelle
  3. propose a next step if learners get stuck: "next-step-guidance" due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to $\mathcal{ISAC}$

- $\mathcal{ISAC}$ is designed for “pure” mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” in Isabelle
  3. propose a next step if learners get stuck: “next-step-guidance” due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to $\mathcal{ISAC}$

- $\mathcal{ISAC}$ is designed for “pure” mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” in Isabelle
  3. propose a next step if learners get stuck: “next-step-guidance” due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

Introduction to *ISAC*

- *ISAC* is designed for "pure" mathematics":
  1. check user input automatically, **flexibly** and reliably: Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
  2. give explanations on request by learners: All underlying mathematics knowledge is **transparent** due to the "LCF-paradigm" in Isabelle
  3. propose a next step if learners get stuck: "next-step-guidance" due to Lucas-Interpretation.
- Engineers raised additional requirements for STEM (**S**cience, **T**echnology, **E**ngineering and **M**athematics)
- Answers to 4 requirements planned — and discussed here

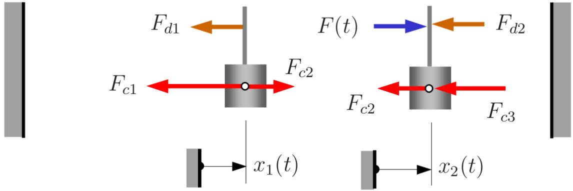
Outline

Figural input

Input of

- arrows at certain locations
- certain direction for arrows
- labels for arrows
- coordinate system with coordinates

to a figure like:



Automated check of input from formal specification

Outline

1. Introduction to $\mathcal{ISAC}$
2. Planned Extensions for STEM Education
   - Input to figural problem description
   - Worksheets with switching levels of abstraction
   - Extend program language with informal justifications
   - Conquer problems by managing sub-problems
3. Conclusions

Symbolic representation

. `Problem` [absorber, 2-mass-oscillator]

1 `Specification`:

2 `Solution`:

21 `Problem` [determine, 2-mass-oscillator, DiffEq]

22 $\begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix} \ddot{x} + \begin{pmatrix} d & 0 \\ 0 & d \end{pmatrix} \dot{x} + \begin{pmatrix} c_1 + c_2 & -c_2 \\ -c_2 & c_1 + c_2 \end{pmatrix} x = \begin{pmatrix} 0 \\ F \end{pmatrix}$

23 `Problem` [solution, 2-mass-oscillator, homogen, DiffEq]

24 $x(t) = \begin{pmatrix} 1 \\ 1 \end{pmatrix} (A_1 \cos \omega_1 t + B_1 \sin \omega_1 t) + \begin{pmatrix} 1 \\ -1 \end{pmatrix} (A_2 \cos \omega_2 t + B_2 \sin \omega_2 t)$,

25 `Problem` [particular, solution, 2-mass-oscillator, DiffEq]

26 $x_1(t) = \begin{pmatrix} 0 \\ a_1 \end{pmatrix} \sin \Omega t, \;\; x_2(t) = \begin{pmatrix} 0 \\ a_2 \end{pmatrix} \sin \Omega t, \;\; a_1 = \frac{F_0 c_2}{(c_1+c_2-m\Omega^2)^2 - c_2^2}, \;\; a_2 = \frac{F_0(c_1+c_2-m\Omega^2)}{(c_1+c_2-m\Omega^2)^2 - c_2^2}$

27 `Problem` [complete, solution, 2-mass-oscillator, DiffEq]

28 $x(t) = \begin{pmatrix} 1 \\ 1 \end{pmatrix} (A_1 \cos \omega_1 t + B_1 \sin \omega_1 t) + \begin{pmatrix} 1 \\ -1 \end{pmatrix} (A_2 \cos \omega_2 t + B_2 \sin \omega_2 t) + \begin{pmatrix} 0 \\ a_1 \end{pmatrix} \sin \Omega t,$

$a_1 = \frac{F_0 c_2}{(c_1+c_2-m\Omega^2)^2 - c_2^2}, \;\; a_2 = \frac{F_0(c_1+c_2-m\Omega^2)}{(c_1+c_2-m\Omega^2)^2 - c_2^2}$

29 `Problem` [compute, spring]

2a $c_2 = 1.2345 \; N$

. $c_2 = 1.2345 \; N$

Numeric representation

. `Problem` [absorber, 2-mass-oscillator]

1 `Specification`:

2 `Solution`:

21 `Problem` [determine, 2-mass-oscillator, DiffEq]

22 $[2\ddot{x}_1 + 0.4\dot{x}_1 + 3.3x_1 - 0.22x_2 = 0, \;\; 2\ddot{x}_2 + 0.4\dot{x}_2 - 0.22x_1 + 3.3x_2 = 0.6]$

23 `Problem` [solution, 2-mass-oscillator, homogen, DiffEq]

24 $[x_1(t) = 0.05e^{-0.1t}(\cos 0.81t + 3.85 \sin 0.81t),$
$x_2(t) = 0.05e^{-0.1t}(\cos 0.81t + 3.85 \sin 0.81t)]$

25 `Problem` [particular, solution, 2-mass-oscillator, DiffEq]

26 $[x_1(t) = -0.05e^{-0.1t}0.59 \sin 1.69t, \;\; x_2(t) = 0.05e^{-0.1t}0.59 \sin 1.69t]$

27 `Problem` [complete, solution, 2-mass-oscillator, DiffEq]

28 $[x_1(t) = 0.05e^{-0.1t}(\cos 0.81t + 3.85 \sin 0.81t - 0.59 \sin 1.69t),$
$x_2(t) = 0.05e^{-0.1t}(\cos 0.81t + 3.85 \sin 0.81t + 0.59 \sin 1.69t)]$

29 `Problem` [compute, absorber]

2a $c_2 = 1.2345\ N$

. $c_2 = 1.2345\ N$

Outline

Extend programs with informal justifications

```
.    partial_function diffeq_2_mass_oscil (m, l_0, [c_1, c_2],
                                  d, springs, dampers, sums) =
1      let
11       begin_parallel
1101         springs = Take springs "forces of springs"
111        parallel
1111         dampers = Take dampers "forces of dampers"
112        parallel
1121         sums = Take sums "mass times acceleration equals sur
12       end_parallel
13       diffeq = Take sums ""
14       diffeq = Substitute [ springs, dampers ]
15       diffeq = Rewrite_Set normalise
16       diffeq = Rewrite_Set vectorify "switch to vector representati
2      in
21       diffeq
```

More Expressive Worksheets

21 `Problem` [determine, 2-mass-oscillator, DiffEq]:

211 `Specification`:

212 `Solution`:

2121 forces of springs

2122 $[F_{c1} = c_1 x_1,\ F_{c2} = c_2(x_2 - x_1),\ F_{c3} = c_1 x_2]$

2123 forces of dampers

2124 $[F_{d1} = d\dot{x}_1,\ F_{d2} = d\dot{x}_2]$

2125 mass times acceleration equals sum of all forces

2126 $[m\ddot{x}_1 = -F_{c1} + F_{c2} - F_{d1},\ \ m\ddot{x}_2 = -F_{c2} - F_{c3} - F_{d2} + F]$

2127 `Substitute` $[F_{c1}, F_{c2}, F_{c3}, F_{d1}, F_{d2}]$

2128 $[m\ddot{x}_1 = -c_1 x_1 + c_2(c_2 - x_1) - d\dot{x}_1,\ \ m\ddot{x}_2 = -c_2(c_2 - x_1) - c_1 x_2 - d\dot{x}_2 + F]$

2129 `Rewrite_Set` normalise

212a $[m\ddot{x}_1 + d\dot{x}_1 + c_1 x_1 - c_2(x_2 - x_1) = 0,\ \ m\ddot{x}_2 + d\dot{x}_2 + c_2(x_2 - x_1) + c_1 x_1 = F]$

212b switch to vector representation

212c $\begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix} \begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{pmatrix} + \begin{pmatrix} d & 0 \\ 0 & d \end{pmatrix} \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} + \begin{pmatrix} c_1 + c_2 & -c_2 \\ -c_2 & c_1 + c_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ F \end{pmatrix}$

22 $\begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix} \ddot{x} + \begin{pmatrix} d & 0 \\ 0 & d \end{pmatrix} \dot{x} + \begin{pmatrix} c_1 + c_2 & -c_2 \\ -c_2 & c_1 + c_2 \end{pmatrix} x = \begin{pmatrix} 0 \\ F \end{pmatrix}$

Outline

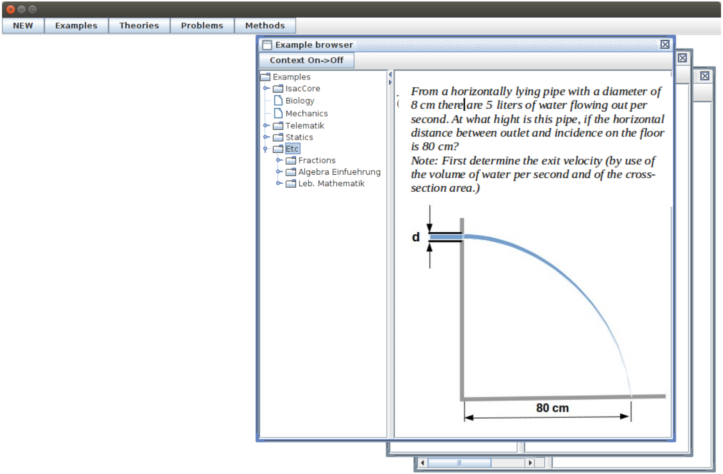
Conquer problems by managing sub-problems

General aspects in problem solving:

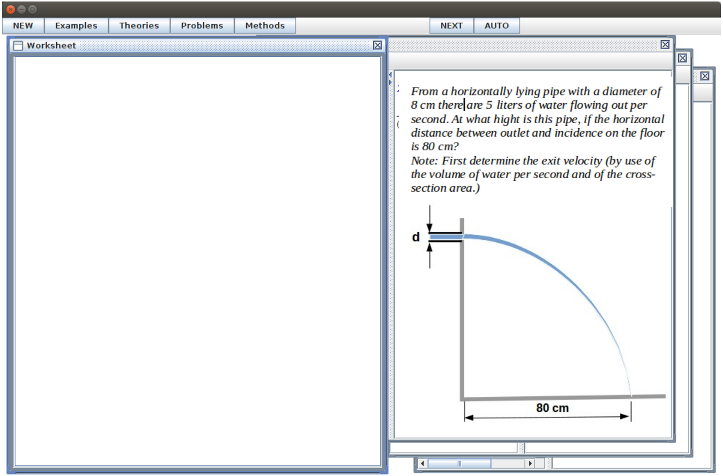
- Aspect 1: Review knowledge for relevance
- Aspect 2: Select relevant knowledge
- Aspect 3: Arrange subproblems to a sequence
- Aspect 4: Connect the subproblems
- Aspect 5: Accomplish unit conversions
- Aspect 6: Solve the subproblems

"Aspect" because not tackled sequentially.

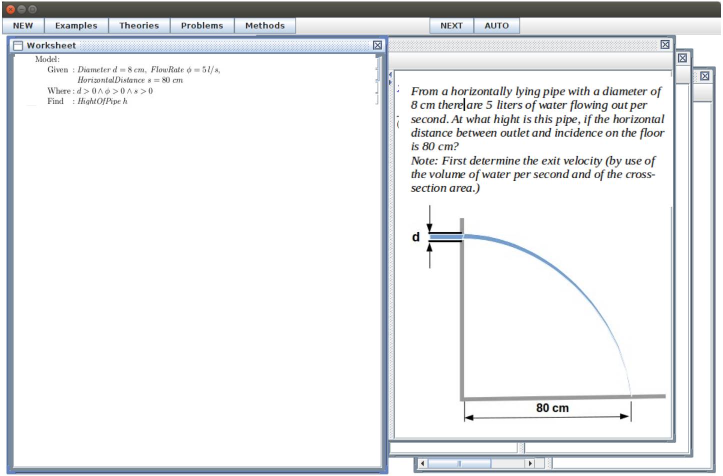
Start Example



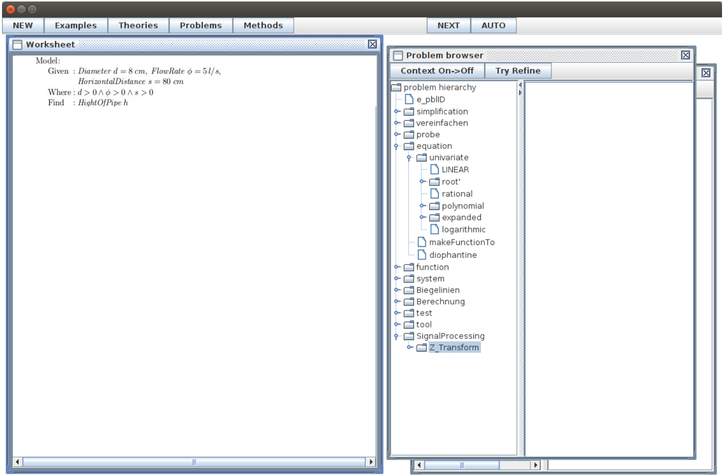
Start Example



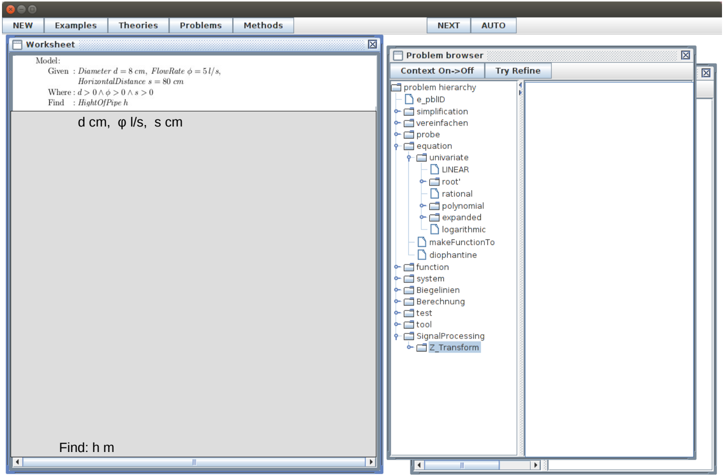
Modelling Phase finished



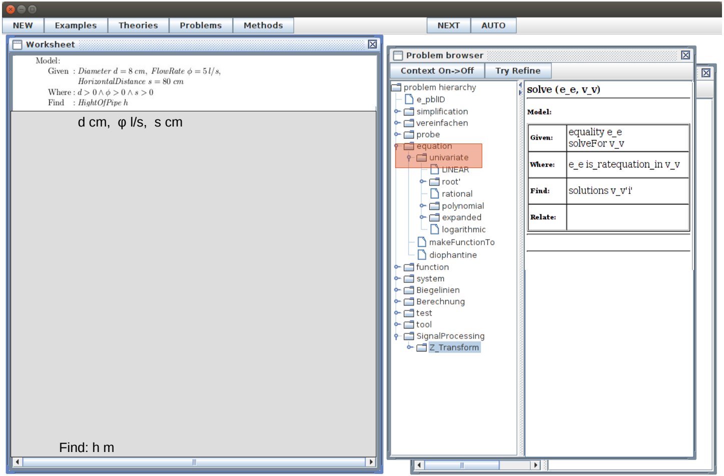
Start Specification Phase



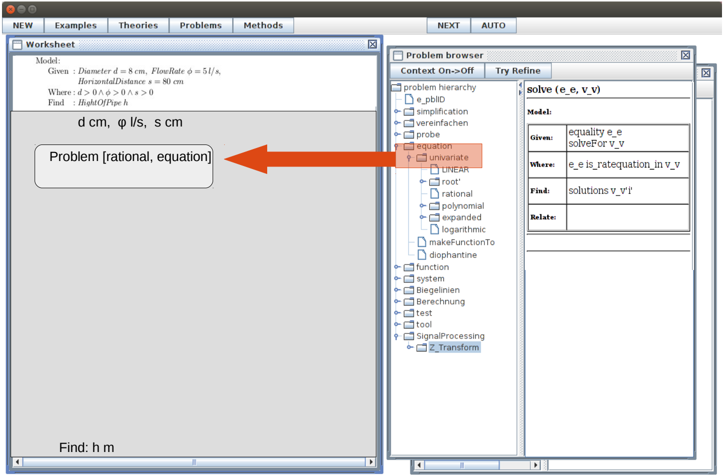
Start Specification Phase



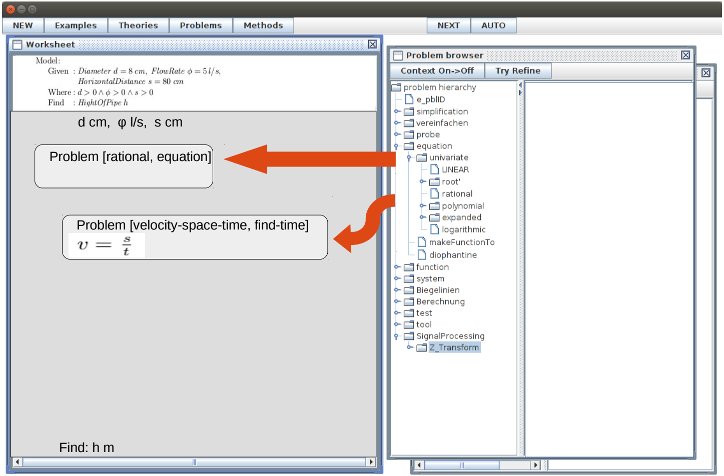
Aspect 1: relevant knowledge assumed to be present



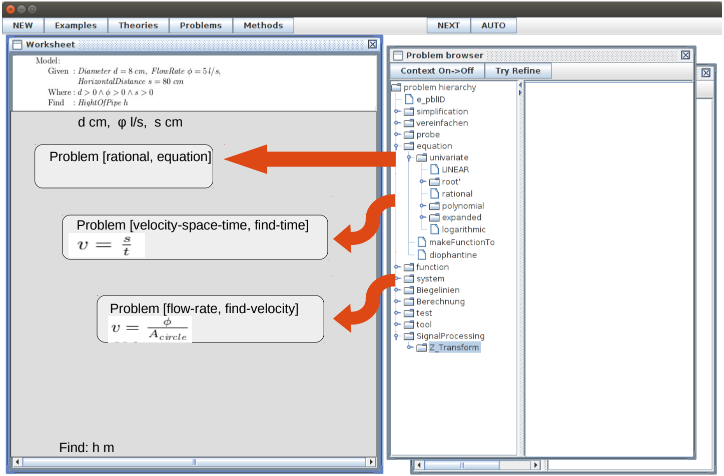
Aspect 2: select knowledge



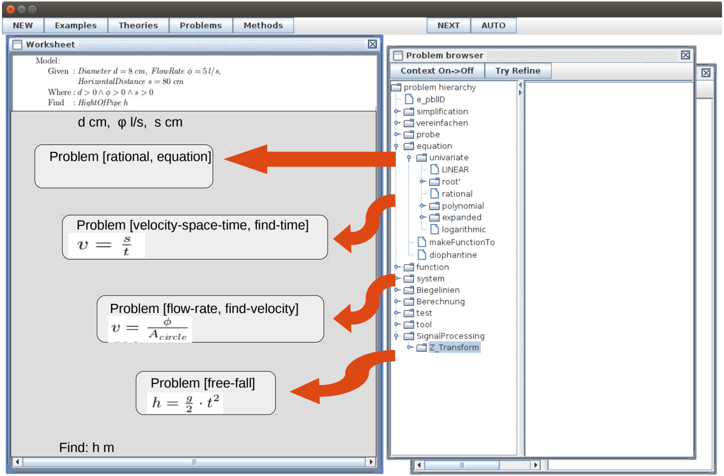
Aspect 2: select relevant knowledge



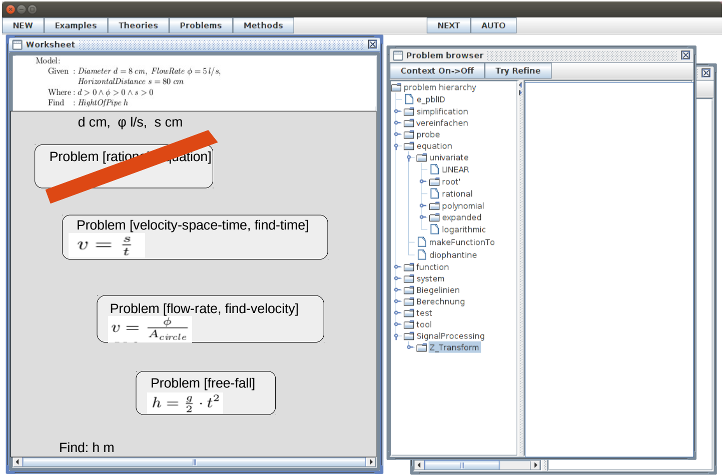
Aspect 2: select relevant knowledge



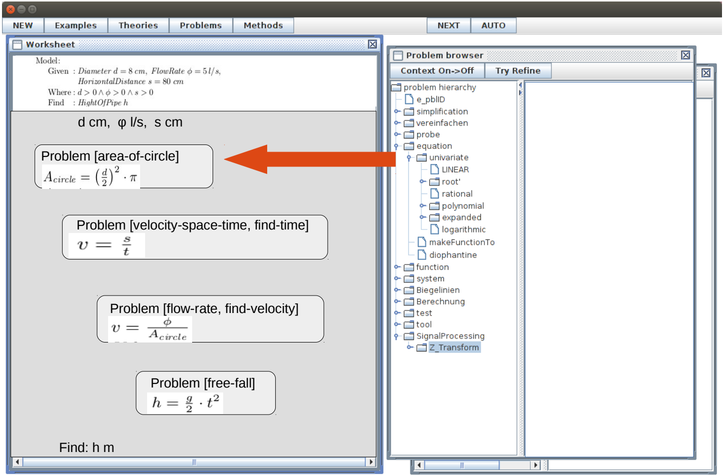
Aspect 2: select relevant knowledge



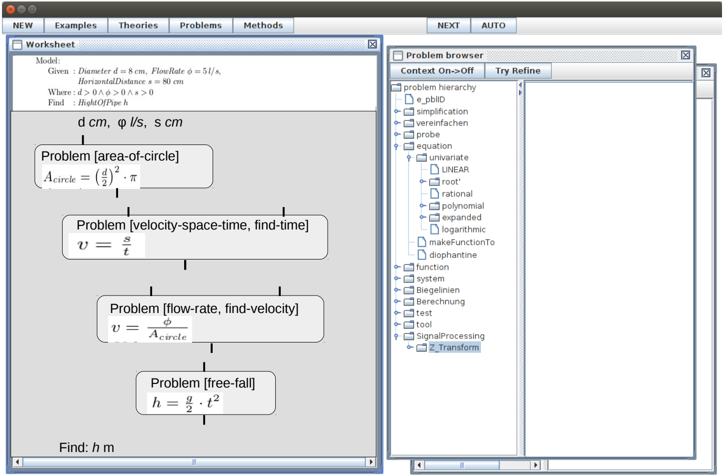
Aspect 2: delete irrelevant knowledge



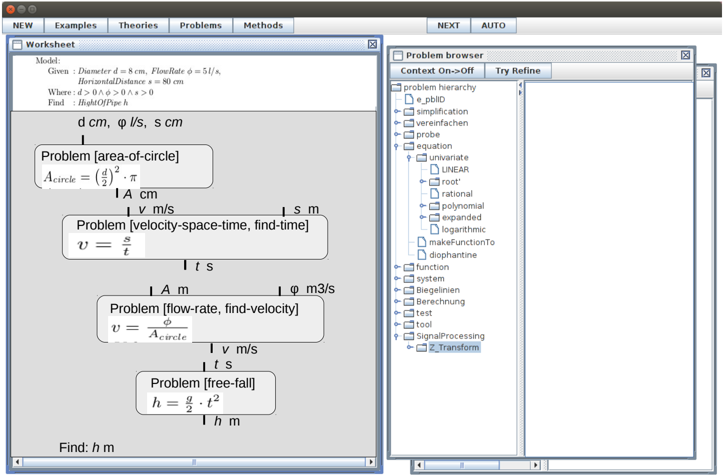
Aspect 2: select relevant knowledge



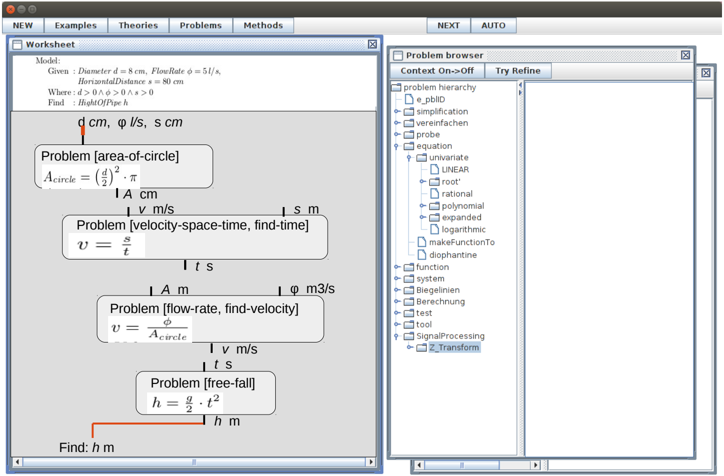
Aspect 3: what is given, what has to be found?



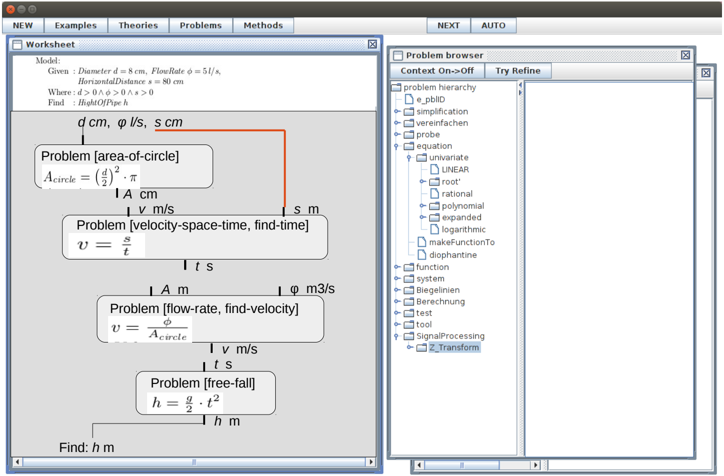
Aspect 3: what is given, what has to be found?



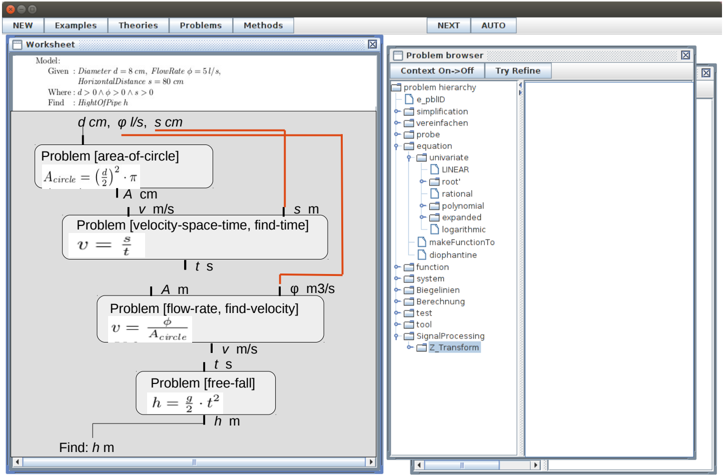
Aspect 3: try to connect "Given" and "Find"



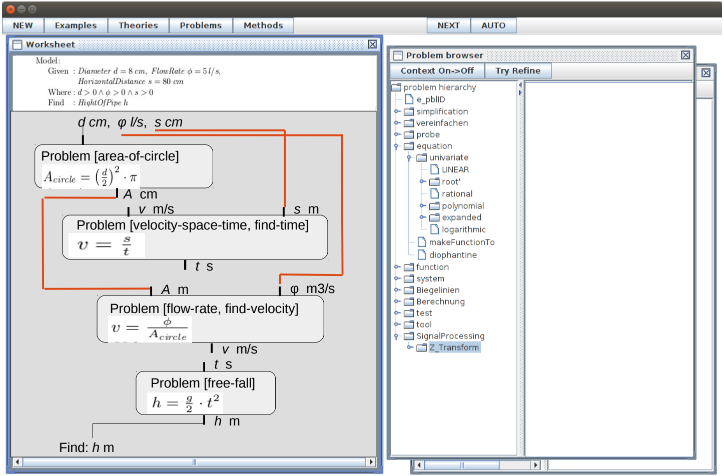
Aspect 3: try to connect "Given" and "Find"



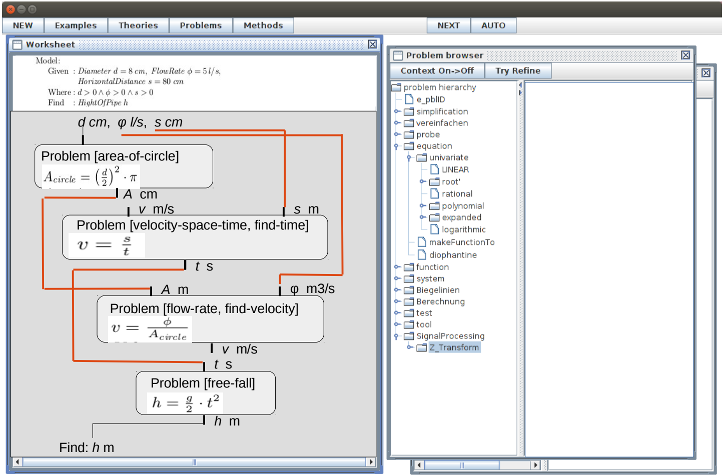
Aspect 3: try to connect "Given" and "Find"



Aspect 3: try to connect "Given" and "Find"

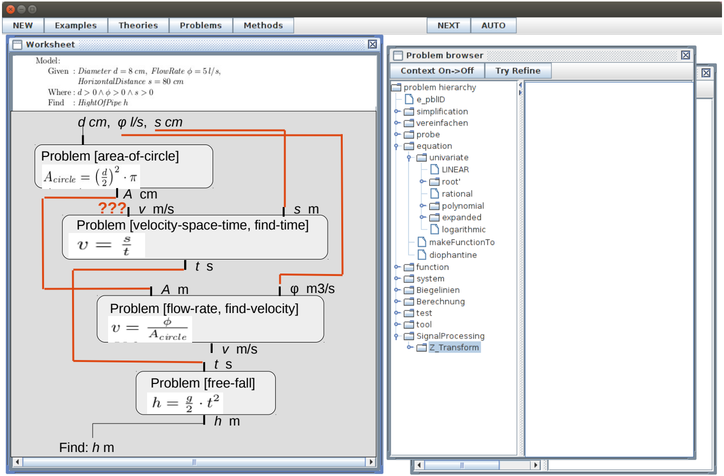


Aspect 3: try to connect "Given" and "Find"

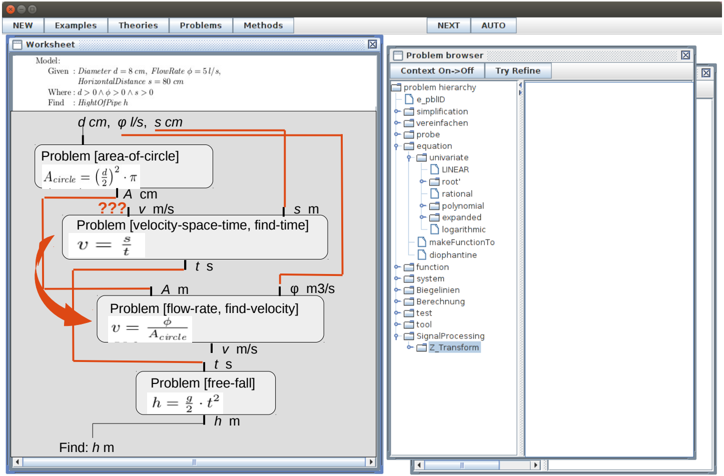


Aspect 3: dangling connection

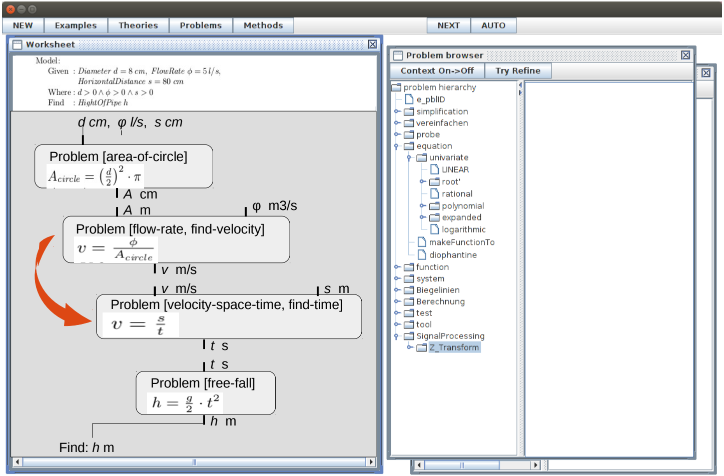
???



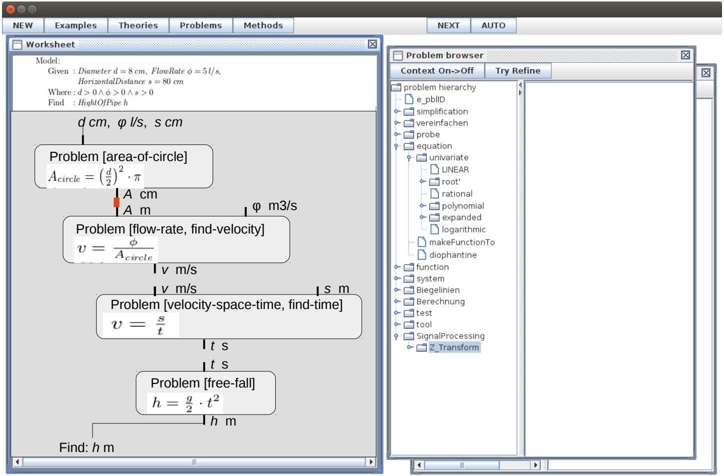
Aspect 3: try another sequence



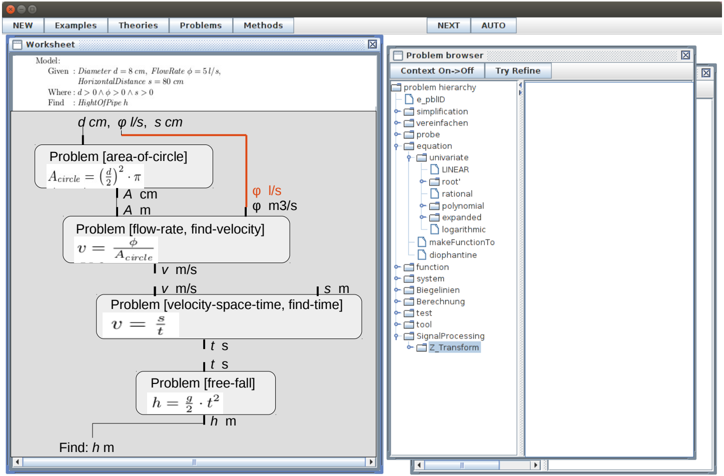
Aspect 3: flipped two subproblems



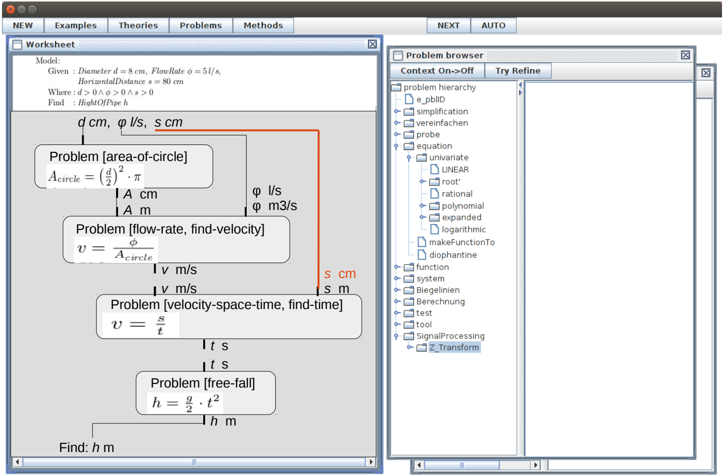
Aspect 3: try to connect “Given” and “Find”



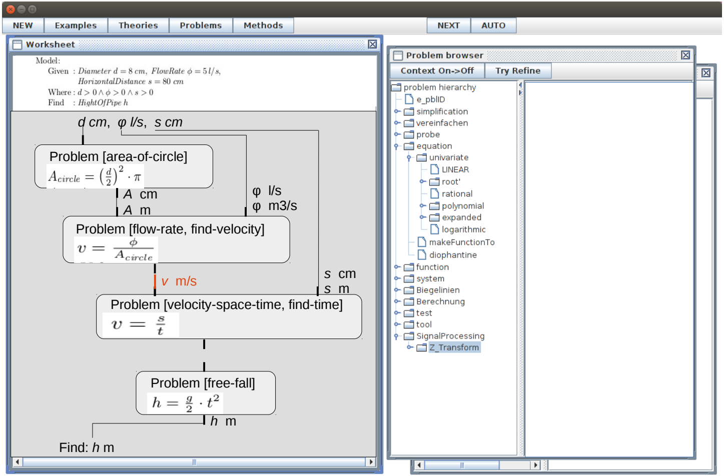
Aspect 3: try to connect "Given" and "Find"



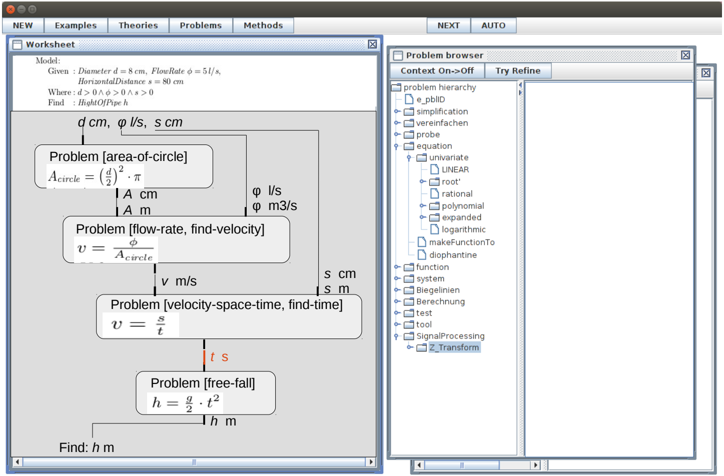
Aspect 3: try to connect “Given” and “Find”



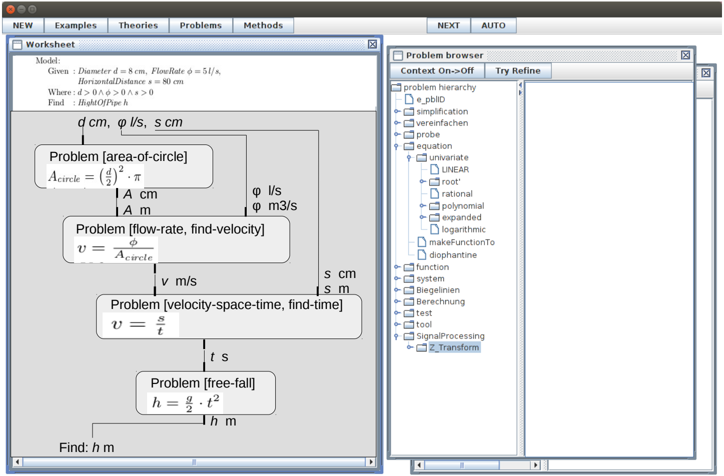
Aspect 3: try to connect "Given" and "Find"



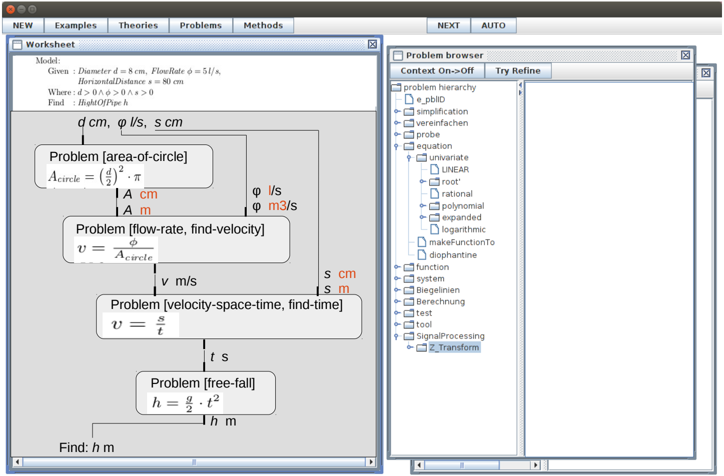
Aspect 3: try to connect “Given” and “Find”



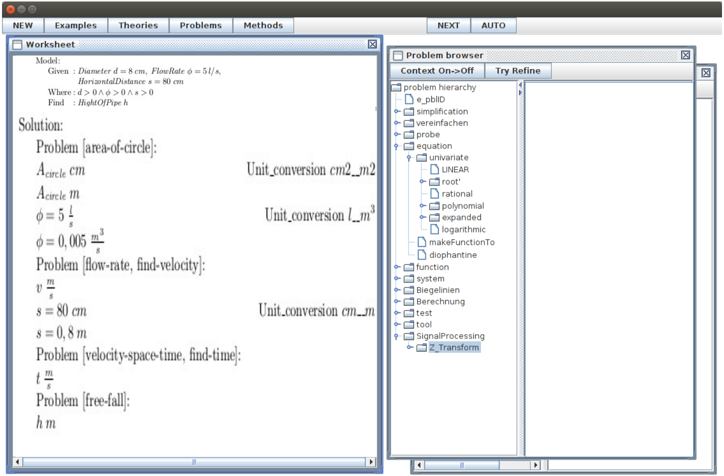
Aspect 3: all connections finished



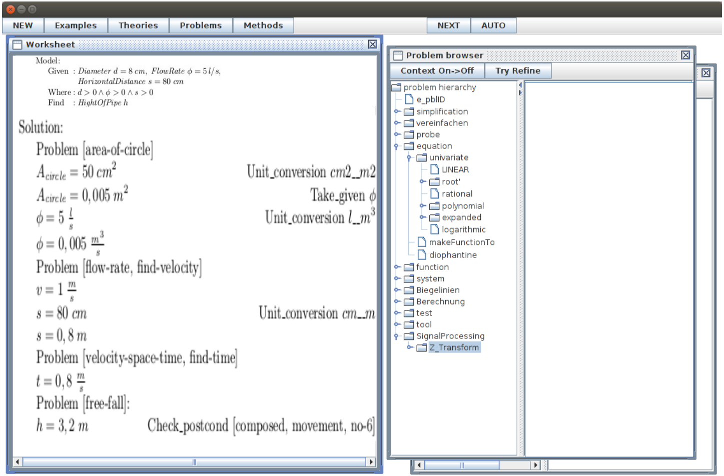
Aspect 4: care about unit conversions



Transition to Solving Phase: units only



Solving Phase finished with complete calculation



Conclusions

- TP-technology is a powerful base for educational SW
  - student can rely on “yes” | “no” from system
  - powerful type systems
  - is still open source
  - quickly growing body of mechanised math
  - “self-referentiality” can foster abstraction
- $\mathcal{ISAC}$ can be extended for STEM with reasonable effort
- . . . as soon as respective knowledge is mechanised:
  ! promotion in cooperation with FM !
- Hope for co-workers and analogous projects
  (in addition to R.J.Back)

Conclusions

- TP-technology is a powerful base for educational SW
  - student can rely on “yes” | “no” from system
  - powerful type systems
  - is still open source
  - quickly growing body of mechanised math
  - “self-referentiality” can foster abstraction
- $\mathcal{ISAC}$ can be extended for STEM with reasonable effort
- ... as soon as respective knowledge is mechanised:
  ! promotion in cooperation with FM !
- Hope for co-workers and analogous projects
  (in addition to R.J.Back)

Conclusions

- TP-technology is a powerful base for educational SW
  - student can rely on "yes" | "no" from system
  - powerful type systems
  - is still open source
  - quickly growing body of mechanised math
  - "self-referentiality" can foster abstraction
- $\mathcal{ISAC}$ can be extended for STEM with reasonable effort
- . . . as soon as respective knowledge is mechanised:
  ! promotion in cooperation with FM !
- Hope for co-workers and analogous projects
  (in addition to R.J.Back)

Conclusions

- TP-technology is a powerful base for educational SW
  - student can rely on “yes” | “no” from system
  - powerful type systems
  - is still open source
  - quickly growing body of mechanised math
  - “self-referentiality” can foster abstraction
- $\mathcal{ISAC}$ can be extended for STEM with reasonable effort
- . . . as soon as respective knowledge is mechanised: ! promotion in cooperation with FM !
- Hope for co-workers and analogous projects (in addition to R.J.Back)

Conclusions

- TP-technology is a powerful base for educational SW
  - student can rely on “yes” | “no” from system
  - powerful type systems
  - is still open source
  - quickly growing body of mechanised math
  - “self-referentiality” can foster abstraction
- $\mathcal{ISAC}$ can be extended for STEM with reasonable effort
- . . . as soon as respective knowledge is mechanised: ! promotion in cooperation with FM !
- Hope for co-workers and analogous projects (in addition to R.J.Back)

Thank you for Attention!