Rigor of TP in Educational Engineering Software

Walther Neuper
University of Technology, Graz, Austria
wneuper@ist.tugraz.at

IICM, Institute for Computer Media

The discipline of Computer Theorem Proving (TP) distinguishes itself by formal rigor in doing mathematics in various application domains [1]. This short paper is, however, *not* on TP but on educational *software based on TP* components. Such software promises advantageous features [7] some of which are demonstrated by a prototype [3] called Isac. Isac is based on the TP Isabelle [2] and generates dialogues similar to interaction with chess software: moves in chess are considered as rigorous formal as steps in calculations are when solving problems in engineering disciplines. Isac checks input of students by use of Isabelle's automated provers, which in turn are provided with necessary logical context by Lucas-Interpretation [6]. This interpreter also allows to propose next steps towards a solution, so roles can be arbitrarily switched between student and system.

This paper reports work in progress in cooperation with universities of applied sciences in Austria. The work concerns a feasibility study on how Isac could serve in engineering education at these universities. Since Isac has been designed for "pure" mathematics, the study encounters several challenges. Below one running example presents three major challenges for discussion; the example is from [9] and slightly changed for reasons discussed in §2:

> Given is a system with two oscillating masses, $m = 2\ kg$, connected by linear springs with length $l_0 = 0.3\ m$ and damped with $d = 0.4\ \frac{Ns}{m}$ as shown in Fig.1. The respective spring constants are $c_1 = 0.11\ \frac{N}{m}$ and $c_2 = 0.22\ \frac{N}{m}$. The masses are located such that $x_1 = x_2 = 0$ with relaxed springs; initially the masses are dislocated with $x_1 = x_2 = 0.05\ m$ and have velocities $v_1 = 0.1\ \frac{m}{s}$ and $v_2 = 0.2\ \frac{m}{s}$ respectively. The right mass is excited by force $F = 0.6 \sin(3t)\ N$. Change the given spring constant c_2 such that the left mass becomes a vibration absorber for the right one (i.e. make the masses oscillate in opposite directions such that the system shows no vibration to the outside).

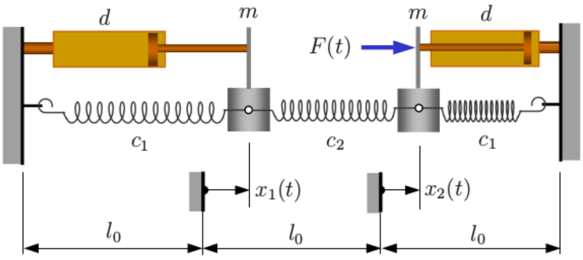

Figure 1: System with two oscillating masses ©W.Steiner 2015 [9]

A solution for c_2 involves modelling the system and comprises several sub-problems: determine the differential equation, solve the homogeneous part, determine the particular solution for F and finally calculate c_2. Below we start with the first subproblem and demonstrate the first challenge raised by geometric descriptions typical for modelling physical systems.

In: A. Editor, B. Coeditor (eds.): Proceedings of the XYZ Workshop, Location, Country, DD-MMM-YYYY, published at http://ceur-ws.org

1 Formal Specification and Geometric “Intuition”

A formal specification in the sense of [4] makes the input `Given` and the output `Find` to a system’s model explicit as shown below; it restricts input by a pre-condition `Where` and relates input with output by a post-condition `Relate`. The first subproblem of the running example is formally specified as follows:

```
21 Problem [determine, 2-mass-oscillator, DiffEq]:
211  Specification:
2111   Model:
21111    Given:  Masses $m = 2\,kg$, Length $l_0 = 0.3\,m$, Consts $\{c_1 = 0.11\,\frac{N}{m},\ c_2 = 0.22\,\frac{N}{m}\}$, Damper $d = 0.4\,\frac{Ns}{m}$
21112    Where:
21113    Find:   Matrixes $\{M(m),\ D(d),\ C(c_1, c_2)\}$, DiffEq $M \cdot \ddot{x} + D \cdot \dot{x} + C \cdot x = F$
21114    Relate: $\exists x.\ \forall t.\ \ t > 0 \Rightarrow M \cdot \ddot{x} + D \cdot \dot{x} + C \cdot x = F$
2112   References:
212  Solution:
```

The `Problem` in line `21` is named such, that a reference into Isac’s knowledge base is given [1]. The other `References` addressed by line `2112` point to a theory, which imports language elements like $\ddot{x}$, and to a method which can create a `Solution`, are collapsed here (as well as the pre-conditions in `Where`). The notation $M(m)$ establishes a literal connection between `Given` and `Find`; the notation is up to discussion. The post-condition in `21114` comprises an $\exists$ not relevant for engineers and might be omitted.

In interactive construction of a `Solution` the challenge for students is to relate forces, for instance

$$m\ddot{x}_1 = -F_{c1} + F_{c2} - F_{d1}, \quad m\ddot{x}_2 = -F_{c2} - F_{c3} - F_{d2} + F(t)$$

And for the task of relating the forces, figures like Fig.2 are used to capture coordinates and forces. Now the problem with Isac’s design is, that such figures capture relations in a precise representation, but this representation is geometric, not formal — and Isac is designed to work with formulas (which would be clumsy in capturing geometric structure here), which can be handled by Isabelle’s components in the background. So the section’s headline advocates “intutition” as opposed to formal specification.

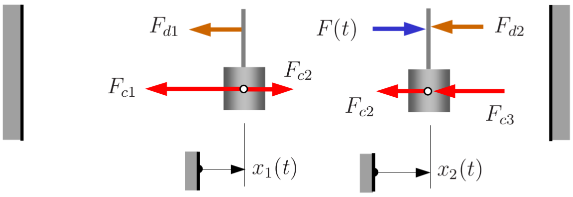

Figure 2: Forces on the oscillating masses ©W.Steiner 2015 [9]

Such figural representations are used frequently and in different engineering disciplines, not only in mechanical engineering. So efforts seem well invested to tackle this design challenge, to sustain Isac’s claim to be a “system that explains itself” and to develop a generally usable component for that purpose.

Such a component shall allow to add coordinates, arrows and associated identifiers at certain positions in a figure. This component also shall provide feedback automatically generated from a formalisation, which has been prepared for each example by Isac’s math-authors. Generation of figural representations like Fig.2 shall become another duty of math-authors (while managing interaction is concern of the component).

2 Learning by Switching Levels of Abstraction

Learning to comprehend abstraction requires experiencing a multitude of **concrete examples** — a fact experienced in the practice of education in general, not only in engineering education and with abstract models in mechanics. Mathematical abstractions, like differential equations, however, have an advantage: they can be computed with concrete values, they even can be dynamically simulated given such values.

Another characteristics of complex learning processes, like comprehension of abstract models, is that they succeed **not in one go**. Learning happens in phases in the brain, which usually are separated by latency periods

[1] `http://www.ist.tugraz.at/projects/isac/www/kbase/pbl/index_pbl.html`

– and these cannot be planned from outside an individual brain; so, a lecture on the behaviour of two oscillating masses (e.g. [9].p.122–129) is only a part of respective learning processes.

So, how to cope with these challenges in learning to comprehend abstraction? Good old LaTeX provided wide margins for personal notes in papers and textbooks again and again; interactive media can do better nowadays, if they are designed appropriately. And it appears obvious: the more such media cover the process of problem solving, the better. Isac claims to cover the whole process; §1 showed, how problem specification is covered by Isac. Below is shown, what else can be done.

Include creation of models into concrete examples

as done with the running example: The problem statement on p.1 contains a concrete request for a particular value of c_2 (below folded into `Specification` in order to save space) — nevertheless the `Solution` should comprise the creation of the underlying abstract model as follows:

. `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 $c_2 = 1.2345\ N$

. $c_2 = 1.2345\ N$

Since above the modelling process is included into the `Solution`, students are enabled, not just to advocate some formula from somewhere (which seduces to use formal models without understanding). A student can regard the (sub-)`Problem`s as black-boxes, of course (if not forced by Isac's dialog guide to actively do the step). But Isac is designed to elicit experimentation, for instance to experiment with different input values to a problem in order to approximate a solution by trial and error. Since solving the above problem with trials soon turns out hopeless, a student might be motivated to make the sub-`Problem`s white-boxes, look into them, study details and to rework creation of the abstract model.

Switch levels of abstraction:

In the above calculation the concrete result $c_2 = 1.2345\ N$ was possible, because the `Specification` contains the concrete values given on p.1, which were input at the beginning of the calculation:

. `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      $c_2 = 1.2345\ N$
.  $c_2 = 1.2345\ N$
```

Note that above also intermediate results are numeric values (copied from a Mathematica notebook): both representation, symbolic and numeric, have their advantages: the first tells about the structure of the model, the latter tells about concrete results (which can be observed in dynamic simulation of the enclosed differential equations, ultimately, see [5]).

Switching between symbolic representation and numeric representation can be done by computer software: this novel feature seems of utmost importance for learning, so it shall be included to Isac within the next development phase. Realisation requires to extend Isac's Lucas-Interpreter [6] such that the interpreter's environment not only takes identifier-value pairs, but associates identifiers with lists of values.

Both design features together, the feature to include models into concrete examples and the feature to switch levels of abstraction, lead to radically new learning scenarios

- The student is offered to review the construction of the abstract model any time (every example concerning, for instance, the two-mass-oscillator includes the respective model: this is accomplished by copy& past for some sub-problems in the respective program [8]).
- The student is not bothered by unwanted details: he or she can skip in a calculation whatever steps they want to (if the dialog [8] allows to do so, which would not be the case in exams, for example).
- Even students of introductory courses can be offered to look into advanced examples like the two-mass-oscillator under consideration: the dialog jumps to some subproblem in the calculation, which is up to exercise in the course (e.g. differentiation, equation solving, etc). Then the student interactively works on the respective subproblem, and if the problem is solved, the system finished the calculation automatically. This way questions like "What for do we learn this method" are anticipated in an unobtrusive way.

3 Formal Deduction and Physical Arguments

Isac is designed such that a student can rely on the system, that wrong steps in a calculation are rejected reliably. From the assumptions (the input in `Given` and the pre-conditions in `Where`) the steps are formally deduced such that finally the post-condition (partially represented in `Relate`) can automatically be proven; for details see [6].

Now, in §1 we encountered an example, where essential assumptions are given by geometric structures (arrows as forces in a figure) and not by formulas. So it seems straightforward, to add informal arguments to the steps of formal deduction. We come back to this sub-`Problem` and proceed from the `Specification` (folded in below) to the `Solution`:

```
21    Problem [determine, 2-mass-oscillator, DiffEq]:
211     Specification:
212     Solution:
2121                                                     forces of springs
2122      $[F_{c1} = c_1x_1,\ F_{c2} = c_2(x_2 - x_1),\ F_{c3} = c_1x_2]$
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      $[m\ddot{x}_1 = -c_1x_1 + c_2(c_2 - x_1) - d\dot{x}_1,\ \ m\ddot{x}_2 = -c_2(c_2 - x_1) - c_1x_2 - d\dot{x}_2 + F]$
2129                                               Rewrite_Set normalise
212a      $[m\ddot{x}_1 + d\dot{x}_1 + c_1x_1 - c_2(x_2 - x_1) = 0,\ \ m\ddot{x}_2 + d\dot{x}_2 + c_2(x_2 - x_1) + c_1x_1 = F]$
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    $\begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix} \ddot{x} + \begin{pmatrix} d & 0 \\ 0 & d \end{pmatrix} \dot{x} + \begin{pmatrix} c_1 + c_2 & -c_2 \\ -c_2 & c_1 + c_2 \end{pmatrix} x = \begin{pmatrix} 0 \\ F \end{pmatrix}$
```

Above the numbers at the left do not belong to an Isac calculation, they are for referencing only. In the middle there are the formulas of the calculation (with tree-like indentation). On the right there are the justifications for the formulas.

In the lines `2127` and `2129` above there are tactics which contribute to formal deduction in the `Solution`. But the lines `2121`, `2123` and `2125` do not contribute to the formal semantics, they are arguments addressing intuitive understanding of physical aspects of modelling. In line `212b` there is even a non-physical argument, useful just for structuring the calculation.

The two latter kinds of arguments are not contained in Isac's original design. But now requirements analysis suggests to extend Isac's programming language with an additional tactic for textual arguments; this extension seems simple. For details, how Isac guides the student during step-wise construction of a calculation and how this guidance is generated automatically, see [8].

Conclusions

The paper demonstrated examples from mechanics in order to show, that Isac's original design is appropriate for education also in engineering disciplines. Requirements analysis in cooperation with experts in engineering education identified additional requirements, which appear realizable with reasonable effort due to TP's power. In particular, Isac as a prototype for TP-based educational systems seems ready for the following three extensions, which have been introduced in §1, §2 and §3 respectively:

1. An interactive graphics-component for input of coordinates, arrows and associated identifiers at certain positions in a figure, while correctness is checked by use of hidden "formalisations" already present in Isac — this shall support geometric intuition in creating physical models.

2. Extension of Isac's Lucas-Interpreter such that the interpreter's environment not only takes identifier-value pairs, but associates identifiers with lists of values (symbolic and numeric) — this shall support comprehending abstract models by switching levels of abstraction.

3. An additional tactic, which plays no role in formal deduction in a calculation, but displays informal arguments for certain steps of calculation — this shall support additional arguments for intuitive understanding.

As soon as these features are implemented, Isac can be considered a "system that explains itself" also for mathematics applied to various engineering disciplines — a system which promises to establish novel learning scenarios in the field.

Acknowledgements

Several experts from the Universities of Applied Sciences in Wels, Hagenberg and Salzburg are involved in the current requirements engineering: Stephan Dreiseitl, Günther Eibl, Klaus Schiefermayr, Wolfgang Steiner and Stefan Sunzenauer. The author owes these persons a great debt of gratitude for their precious time spent and the novel ideas contributed to the design of Isac and to this paper.

Disclaimer

There was no time asking the mentioned experts to review the paper at hand; so any flaws and mistakes in the paper are in full responsibility of the author.

References

[1] *Archive of Formal Proofs.* `http://afp.sourceforge.net`.

[2] *Generic proof assistant "Isabelle".* `http://isabelle.in.tum.de/`.

[3] *Isac-project.* `http://www.ist.tugraz.at/isac/History`.

[4] Dines Bjørner (2006): *Software Engineering. Texts in Theoretical Computer Science* 1,2,3, Springer, Berlin, Heidelberg.

[5] Sarah Lichtblau: *Motion of Two Masses Connected by Springs.* `http://demonstrations.wolfram.com/MotionOfTwoMassesConnectedBySprings/`. Wolfram Demonstrations Project.

[6] Walther Neuper (2012): *Automated Generation of User Guidance by Combining Computation and Deduction.* pp. 82–101, doi:10.4204/EPTCS.79.5. `http://eptcs.web.cse.unsw.edu.au/paper.cgi?THedu11.5`.

[7] Walther Neuper (2013): *On the Emergence of TP-based Educational Math Assistants.* 7, pp. 110–129. Available at `https://php.radford.edu/~ejmt/ContentIndex.php#v7n2`. Special Issue "TP-based Systems and Education".

[8] Walther Neuper (2016): *Lucas-Interpretation from Users' Perspective.* In: *submitted to CICM*, Bialystok, Poland. `http://www.ist.tugraz.at/projects/isac/publ/lucin-user-view.pdf`.

[9] Wolfgang Steiner (2015): *Vorlesungsskriptum Technische Mechanik III.* FH OÖ, Fakultät für Technik und Umweltwissensschaften.