CTP-based Tutoring in Applied Mathematics
Preparing Cooperation in an FP7-ICT Proposal

Walther Neuper

Institute for Computer Media (IICM)
Graz University of Technology

29.Nov.2011
1. CTP = “Computer Theorem Proving”
   CTP — History and Future

   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3. Planning for an FP7-ICT Proposal
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
Outline

1 CTP = “Computer Theorem Proving”
   CTP — History and Future

2 Advances of CTP-based Educational Math Assistants . . .
   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3 Planning for an FP7-ICT Proposal
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
CTP — History and Future

- **CTP’s history as a tool for mathematicians**
  - **1977** L.S. van Benthem Jutting checks *proofs* in Landau’s “Grundlagen” using the AUTOMATH System.
  - **1976** Kenneth Appel and Wolfgang Haken informally check 1.936 maps by computer within a proof of the Four Colour Theorem.
  - **2005** Georges Gonthier provides a computer-checked proof of the *Four Colour Theorem*.

- **CTP’s future: Formal Domain Modeling by engineers?**
  - proving *software* properties reveals design flaws early
  - verification of high-security *software* systems
  - systems for banking, flight control, etc include *hardware*
  - ? Formal Domain Modeling in respective disciplines:
    - ??? in Electrical Engineering …
    - ??? in Structural Engineering …
    - ??? in …
CTP — History and Future

- CTP’s history as a tool for mathematicians
  - **1977** L.S. van Benthem Jutting checks proofs in Landau’s “Grundlagen” using the AUTOMATH System.
  - **1976** Kenneth Appel and Wolfgang Haken informally check 1,936 maps by computer within a proof of the Four Colour Theorem.
  - **2005** Georges Gonthier provides a computer-checked proof of the Four Colour Theorem.

- CTP’s future: Formal Domain Modeling by engineers?
  - proving software properties reveals design flaws early
  - verification of high-security software systems
  - systems for banking, flight control, etc include hardware
  - ? Formal Domain Modeling in respective disciplines:
    - ??? in Electrical Engineering …
    - ??? in Structural Engineering …
    - ??? in …
CTP — History and Future

- CTP’s history as a tool for mathematicians
  - 1977 L.S. van Benthem Jutting checks proofs in Landau’s “Grundlagen” using the AUTOMATH System.
  - 1976 Kenneth Appel and Wolfgang Haken informally check 1.936 maps by computer within a proof of the Four Colour Theorem.
  - 2005 Georges Gonthier provides a computer-checked proof of the Four Colour Theorem.
- CTP’s future: Formal Domain Modeling by engineers?
  - proving software properties reveals design flaws early
  - verification of high-security software systems
  - systems for banking, flight control, etc include hardware
  - ? Formal Domain Modeling in respective disciplines:
    - ??? in Electrical Engineering …
    - ??? in Structural Engineering …
    - ??? in …
• CTP’s history as a tool for mathematicians
  • 1977 L.S. van Benthem Jutting checks proofs in Landau’s “Grundlagen” using the AUTOMATH System.
  • 1976 Kenneth Appel and Wolfgang Haken informally check 1.936 maps by computer within a proof of the Four Colour Theorem.
• 2005 Georges Gonthier provides a computer-checked proof of the Four Colour Theorem.
• CTP’s future: Formal Domain Modeling by engineers?
  • proving software properties reveals design flaws early
  • verification of high-security software systems
  • systems for banking, flight control, etc include hardware
  • ? Formal Domain Modeling in respective disciplines:
    • ??? in Electrical Engineering …
    • ??? in Structural Engineering …
    • ??? in …
CTP — History and Future

• CTP’s history as a tool for mathematicians
  • **1977** L.S. *van Benthem Jutting* checks *proofs* in Landau’s “Grundlagen” using the AUTOMATH System.
  • **1976** *Kenneth Appel* and *Wolfgang Haken* informally check 1.936 maps by computer within a proof of the Four Colour Theorem.
  • **2005** *Georges Gonthier* provides a computer-checked proof of the **Four Colour Theorem**.

• CTP’s future: Formal Domain Modeling by *engineers*?
  • proving *software* properties reveals design flaws early
  • verification of high-security *software* systems
    • systems for banking, flight control, etc include *hardware*
    • ? Formal Domain Modeling in respective disciplines:
      • ??? in Electrical Engineering . . .
      • ??? in Structural Engineering . . .
      • ??? in . . .
• CTP’s history as a tool for mathematicians
  • **1977** *L.S. van Benthem Jutting* checks proofs in Landau’s “Grundlagen” using the AUTOMATH System.
  • **1976** *Kenneth Appel* and *Wolfgang Haken* informally check 1.936 maps by computer within a proof of the Four Colour Theorem.
  • **2005** *Georges Gonthier* provides a computer-checked proof of the **Four Colour Theorem**.

• CTP’s future: Formal Domain Modeling by engineers?
  • proving software properties reveals design flaws early
  • verification of high-security software systems
  • systems for banking, flight control, etc include hardware
  • ? Formal Domain Modeling in respective disciplines:
    • ??? in Electrical Engineering . . .
    • ??? in Structural Engineering . . .
    • ??? in . . .
Outline

1. CTP = “Computer Theorem Proving”
   CTP — History and Future

   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3. Planning for an FP7-ICT Proposal
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
Advances in Step-wise Solving

• ... in step-wise solving engineering problems:

• ... in explaining underlying knowledge:

• ... in checking steps input by the student:

• ... in assessing step-wise problem solving:
Advances in Step-wise Solving

• ... in step-wise solving engineering problems: CTP provides a consistent framework for whole solving process.

• ... in explaining underlying knowledge:

• ... in checking steps input by the student:

• ... in assessing step-wise problem solving:
Outline

1. **CTP = “Computer Theorem Proving”**
   CTP — History and Future

2. **Advances of CTP-based Educational Math Assistants . . .**
   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3. **Planning for an FP7-ICT Proposal**
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
Advances in Explaining

- ... in step-wise solving engineering problems: *CTP provides a consistent framework for whole solving process.*

- ... in explaining underlying knowledge:

- ... in checking steps input by the student:

- ... in assessing step-wise problem solving:
Advances ... in Explaining

• ... in step-wise solving engineering problems: 
  *CTP provides a consistent framework for whole solving process.*

• ... in explaining underlying knowledge:
  *CTP has the knowledge transparent in traditional notation.*
  *Context-sensitive access to multi-media content!*

• ... in checking steps input by the student:

• ... in assessing step-wise problem solving:
**Outline**

1. **CTP = “Computer Theorem Proving”**
   - CTP — History and Future

2. **Advances of CTP-based Educational Math Assistants . . .**
   - . . . in Step-wise Solving Engineering Problems
   - . . . in Explaining Underlying Knowledge
   - . . . in Checking Steps Input by Students
   - . . . in Assessing Step-wise Problem Solving

3. **Planning for an FP7-ICT Proposal**
   - Authoring in Engineering Applications
   - Tasks and Partners in the Proposal
Advances . . . in Checking

- . . . in step-wise solving engineering problems: 
  CTP provides a consistent framework for whole solving process.

- . . . in explaining underlying knowledge: 
  CTP has the knowledge transparent in traditional notation.
  Context-sensitive access to multi-media content!

- . . . in checking steps input by the student:

- . . . in assessing step-wise problem solving:
Advances . . . in Checking

- . . . in step-wise solving engineering problems:
  CTP provides a consistent framework for whole solving process.

- . . . in explaining underlying knowledge:
  CTP has the knowledge transparent in traditional notation.

  Context-sensitive access to multi-media content !

- . . . in checking steps input by the student:
  Input creates a proof situation:
  CTP proves derivability of input formula.

- . . . in assessing step-wise problem solving:
Outline

1. CTP = “Computer Theorem Proving”
   CTP — History and Future

   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3. Planning for an FP7-ICT Proposal
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
Advances ... in Assessing

• ... in step-wise solving engineering problems:
  *CTP provides a consistent framework for whole solving process.*

• ... in explaining underlying knowledge:
  *CTP has the knowledge transparent in traditional notation.*
  *Context-sensitive access to multi-media content!*

• ... in checking steps input by the student:
  *Input creates a proof situation:*
  *CTP proves derivability of input formula.*

• ... in assessing step-wise problem solving:
Advances ... in Assessing

- ... in step-wise solving engineering problems:
  \textit{CTP provides a consistent framework for whole solving process.}

- ... in explaining underlying knowledge:
  \textit{CTP has the knowledge transparent in traditional notation.}
  \textit{Context-sensitive access to multi-media content!}

- ... in checking steps input by the student:
  \textit{Input creates a proof situation:}
  \textit{CTP proves derivability of input formula.}

- ... in assessing step-wise problem solving:
  \textit{One and the same CTP-based system accomplishes tutoring and assessment.}
Outline

1. CTP = “Computer Theorem Proving”
   CTP — History and Future

   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3. Planning for an FP7-ICT Proposal
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
Given the Prototype add . . .
Mathematics Authoring
Mathematics Authoring

```
Problem (B, bendl)
Problem (B, load2bl)
Q(x) = c-q.x, M(x) = ...
Problem (B, sidecds)
L.q = x, 0 = c.2+L.c...
solveSys [0=c_3, ...
c = q.L]

Theory
Theorems
Belastung_Querkraft
- q_ ?x = V' ?x
Querkraft_Moment
V ?x = M_b' ?x
Moment_Neigung
M_b ?x = - EI * y'' ?x
Definition
q is_integrable <=> ...

Specification
In:   function q,
       Length L
pre: q is_integrable
       ∆  L > 0
out: function y(x)
Post: y(0)=0
      ∆ y'(0)=0
      ∆ V(0)=q.L
      ∆ M_b(L)=0

Dialog
Lucas-Interpreter
Isabelle
CTP
```
Mathematics Authoring

theory

specification

program

Theorems
Belastung_Querkraft
\( q_\cdot ?x = V' \cdot ?x \)
Querkraft_Moment
\( V \cdot ?x = M_b' \cdot ?x \)
Moment_Neigung
\( M_b \cdot ?x = -EI \cdot y'' \cdot ?x \)
Definition
q is_integrable \( \iff \)

In: function q,
       Length L
pre: q is_integrable
       \( \Delta L > 0 \)
out: function y(x)
Post: \( y(0)=0 \)
      \( \Delta y'(0)=0 \)
      \( \Delta V(0)=q.L \)
      \( \Delta M_b(L)=0 \)

Script B (q, L, v, Cs) =
  LET
  funs = Subproblem (thy, pbl, met) q, L, v
  equs = Subproblem ...
  sols = Subproblem ...
  B = Take (LAST funs)
  B = ((Substitute sols) @
       (Rewrite_Set poly)) B
  IN B

Problem (B, bendl)
Problem (B, load2bl)
\( Q(x) = c-q.x, M(x) = ... \)
Problem (B, sidecds)
\( L.q = x, 0 = c_2+L.c... \)
solveSys \( 0=c_3, ... \)
\( c = q.L \]

Dialog
Lucas-Interpreter
CTP Isa-belle

In:   function q,
       Length L
pre: q is_integrable
       \( \Delta L > 0 \)
out: function y(x)
Post: \( y(0)=0 \)
      \( \Delta y'(0)=0 \)
      \( \Delta V(0)=q.L \)
      \( \Delta M_b(L)=0 \)

Script B (q, L, v, Cs) =
  LET
  funs = Subproblem (thy, pbl, met) q, L, v
  equs = Subproblem ...
  sols = Subproblem ...
  B = Take (LAST funs)
  B = ((Substitute sols) @
       (Rewrite_Set poly)) B
  IN B

Problem (B, bendl)
Problem (B, load2bl)
\( Q(x) = c-q.x, M(x) = ... \)
Problem (B, sidecds)
\( L.q = x, 0 = c_2+L.c... \)
solveSys \( 0=c_3, ... \)
\( c = q.L \]

student
Course Design

**Theory**
- Theorems
  - Belastung_Qerkraft: \( q(x) = V'(x) \)
  - Querkraft_Moment: \( V(x) = M_b'(x) \)
  - Moment_Neigung: \( M_b(x) = -EI \cdot y''(x) \)
- Definition: \( q \) is integrable \( \iff \)

**Specification**
- In: function \( q \), Length \( L \)
- pre: \( q \) is integrable
- \( \Delta L > 0 \)
- out: function \( y(x) \)
- Post: \( y(0) = 0 \), \( \Delta y'(0) = 0 \), \( \Delta V(0) = q \cdot L \), \( \Delta M_b(L) = 0 \)

**Program**
- Script \( B(q, L, v, Cs) = \)
  - LET
    - funs = Subproblem \( \text{thy}, pbl, \text{met} \) \( q, L, v \)
    - equs = Subproblem ...
    - sols = Subproblem ...
  - \( B = \text{Take} \) (LAST funs)
  - \( B = (\text{Substitute sols}@\text{Rewrite_Set poly}) B \)

**Dialog**

**Lucas-Interpreter**

**CTP Isa-belle**

**Student**

**Course Design**

**Examples**

**Media**

**Isa-belle Dialog**

**Mathematics**

**Authoring**

**Course Design**

**Problem (B, bendl)**
- Problem (B, load2bl)
  - \( Q(x) = c - q \cdot x, M(x) = ... \)
- Problem (B, sidecds)
  - \( L, q = x, 0 = c, 2 + L, c ... \)
  - solveSys \( [0 = c_3, ... \]
  - \( c = q \cdot L \)

**FP7-ICT**
- Authoring
- Organization

**CTP**
- History - Future
- Stepwise Solving
- Explain Knowledge
- Check User-Input
- Assessment
Dialog Authoring

mathematics authoring

theory

specification

program

In: function q,
       Length L
pre: q is_integrable
       ∆ L > 0
out: function y(x)
Post: y(0)=0 ∆ y'(0)=0
      ∆ V(0)=qL
      ∆ M_b(L)=0

Script B (q, L, v, Cs) =
LET funs = Subproblem (thy, pbl, met) q, L, v
equs = Subproblem ...
sols = Subproblem ...
B = Take (LAST funs)
B = ((Substitute sols)@
      (Rewrite_Set poly)) B
IN B

Dialog
Lucas-
Interpreter
CTP
Isa-
abelle

Problem (B, bendl)
Problem (B, load2bl)
Problem (B, sidecds)

Problem (B, bendl)
Problem (B, load2bl)
Problem (B, sidecds)

Q(x) = c-q.x, M(x) = ...
L.q = x, 0 = c, 2+L.c...
solveSys [0=c_3, ...]
c = qL

student

course design
dialog authoring

examples
media
dialog rules

R1: spec, ok => i1
R2: spec, error => i2
R3: solve, ok => i1
R4: solve, error => i2
R5: thm, error => i5
Survey Authoring

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter

- add mathematics authoring on
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, …)

- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc

- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Authoring

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter

- add mathematics authoring on
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, ...)

- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc

- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Authoring

• Given the prototype of
  • dialog
  • CTP-based programming language
  • Lucas-Interpreter

• add mathematics authoring on
  • theories: theorems, (proofs), definitions
  • specifications: one per example class
  • programs: one per example class
  • Computer Algebra (equation solving, ...)

• while course design determines
  • example collections: for lecture, lab, etc
  • media for explanations: movies on examples, theorems, problems, etc

• add dialog authoring
  • determines rules for exercises, for assignments
  • adds, removes, modifies rules
Survey Authoring

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter

- add mathematics authoring on
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, . . . )

- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc

- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Roles: ... improved by

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter

- add mathematics authoring on (assist. by Math TUG)
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, ...)

- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc

- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Roles: ... improved by

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter

- add mathematics authoring on (assist. by Math TUG)
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, ...)

- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc

- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Roles: … improved by

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter
- add mathematics authoring on (assist. by Math TUG)
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, …)
- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc
- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Roles: … improved by

- Given the prototype of
  - dialog
  - CTP-based programming language
  - Lucas-Interpreter
- add mathematics authoring on (assist. by Math TUG)
  - theories: theorems, (proofs), definitions
  - specifications: one per example class
  - programs: one per example class
  - Computer Algebra (equation solving, …)
- while course design determines
  - example collections: for lecture, lab, etc
  - media for explanations: movies on examples, theorems, problems, etc
- add dialog authoring
  - determines rules for exercises, for assignments
  - adds, removes, modifies rules
Survey Roles: ... improved by

- Given the prototype of
  - dialog  
  - CTP-based programming language  
  - Lucas-Interpreter  

- add mathematics authoring on  (assist. by Math TUG)
  - theories: theorems, (proofs), definitions  
  - specifications: one per example class  
  - programs: one per example class  
  - Computer Algebra (equation solving, ...)

- while course design determines
  - example collections: for lecture, lab, etc  
  - media for explanations: movies on examples, theorems, problems, etc

- add dialog authoring
  - determines rules for exercises, for assignments  
  - adds, removes, modifies rules
1. CTP = “Computer Theorem Proving”
   CTP — History and Future

   . . . in Step-wise Solving Engineering Problems
   . . . in Explaining Underlying Knowledge
   . . . in Checking Steps Input by Students
   . . . in Assessing Step-wise Problem Solving

3. Planning for an FP7-ICT Proposal
   Authoring in Engineering Applications
   Tasks and Partners in the Proposal
# Tasks and Partners

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Partners</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTP-based cloud services</td>
<td>Université Paris-Sud</td>
</tr>
<tr>
<td>formula-oriented systems</td>
<td>Åbo Akademi University</td>
</tr>
<tr>
<td>CTP-based geometry systems</td>
<td>TUG &amp; RISC Linz</td>
</tr>
<tr>
<td></td>
<td>DFKI</td>
</tr>
<tr>
<td>test design and evaluation</td>
<td>Université de Strasbourg</td>
</tr>
<tr>
<td></td>
<td>Univerzitet u Beogradu</td>
</tr>
<tr>
<td></td>
<td>Universidade de Coimbra</td>
</tr>
<tr>
<td>implementation in education</td>
<td>IICM TUG</td>
</tr>
<tr>
<td></td>
<td>Ellinogermaniki Agogi</td>
</tr>
</tbody>
</table>
Thank you for attention!
Hope for fruitful cooperation!