Seminar: Intelligent Tutoring Systems
(http://www.dfki.de/~horacek/tutor-nl.html)

Time and place: Wednesday, 16-18 seminar room 001, bldg. E1.7
(begin/introduction 26.10.)

Topics include
Interaction with learners, analysis and generation of dialog contributions
models of problem solving, tutoring strategies, major systems,
recognizing emotions and adapting system behavior accordingly

Students interested in the seminar please contact

Helmut Horacek
DFKI old building, 1.25, Tel: 85775-2450
email: helmut.horacek@dfki.de
What is an intelligent tutoring system (ITS)?
What is an intelligent tutoring system (ITS)?

a digitalized teacher
Properties of intelligent tutoring systems

Tutoring task

Categories of human tutoring:
Classroom, one-to-one tutoring

Features of intelligent tutors

Not just a collection of instances of tutor reactions
Abstraction into descriptions of tutorial settings
Mechanisms that mimic aspects of human tutoring
Reasonable handling of unexpected student reactions
Features met by some tutoring systems:
Mixed initiative, interactive learning, instructional modeling, self-improving
Distinctions to Computer Aided Instructions

Computer Aided Instruction

Program designer specifies input and output in terms of terminal behaviors
Material to be learned, problem to be solved, feedback to answer
The teacher constructs all branching in the program ahead of time

"Intelligent" tutoring

"Intelligent means that the system uses inference mechanisms to provide coaching, explanation, or other information to the student performing a task. Further, it implies that this information is tuned to the context of the student's ongoing work and/or a model of the student's evolving knowledge."

Alan Lesgold
Why building and using intelligent tutoring systems?

Benefits of tutoring

Teaching is necessary to acquire knowledge and skills

Constructivist teaching – learning unique for each individual
   Students may make mistakes, query processes, monitor their own progress

Inquiry and social discussion
   essential for critical thinking, problem solving and reasoning

Motivation for ITS

Overcoming the bottleneck of human tutors

Availability of expertise and teaching material

But: operationalization of tutoring capabilities increasingly difficult
GUIDON – the first intelligent tutoring system

Domain of tutoring

Teaching medical knowledge
Classification and problem solving

Size of the system

400 domain rules
200 tutoring rules

Main achievement as an ITS

Explicit separation of domain knowledge and pedagogical knowledge
A SUCCESS STORY - MS. LINDQUIST

Patented system, used in hundreds of US schools

What the system is dedicated to
Algebra Word Problem

Method
based on a videotaped analysis and research on an expert tutor

Some statements by Neal Heffernan (principle author)
I have spent 4 years working on nothing but studying how to teach students to write algebra expressions given a word problem. This task is very important since a student can't make any use of all the algebra symbolic manipulation skills unless she can translate real world problems into the language of algebra.

I have studied how to make tutoring software that will help students learn this skill.

I have studied what make these problems so hard
A tutoring session with Ms. Lindquist (algebra tutor)

T1: Hello. (After presenting a 'bike-trip' problem).
    Please write an expression for the total time the trip took. (symbolize)
S1: Help! (The tutor breaks down the problem into steps)
T2: Let me ask an easier question. Please compute the time actually on the bikes
    if instead the speed was 20 miles per hour, the time of the break was 2
    hours, and the distance ridden was 100 miles?
S2: 7
T3: Correct. What math did you do to get that 7?
S3: 2+5 (Student does not completely articulate)
T4: Okay. But how did you get the 5? (similar to T3)
S4: 100/20
T5: Correct. Now lets try this again, what math did you do to get the 7?
    (same question as T3 but inserting the phrase 'lets try this again')
S5: 2+100/20 (correct articulation)
T6: Correct. Now write your answer of '2+100/20' using the variables given in
    the problem. (generalize like T4 from previous example)
S6: b+m/s
T7: Correct.
Some tutoring skill incorporated in Ms. Lindquist

Teaching techniques – implementing human strategies

Partitioning/simplifying the problem
Elaborating instantiations
Combining and abstracting partial solutions

Communicative behavior

Encouraging attitude
Accepting partially correct answers
Emphasizing discourse coherence

Innovations

Feedback not based on explicit buggy rules
Chains of hints as progressively more explicit messages
IMPORTANCE OF NATURAL LANGUAGE

Empirical investigations

Natural language communication makes tutoring effective

[Moore 2000]

Presenting reasoning structures in natural language increases subjects' performance

[Di Eugenio et al. 2002]

Natural language evaluative arguments convey object preferences more effectively

[Carenini, Moore 2001]

Preference of natural language variant shown to be statistically significant
Building an ITS is an interdisciplinary activity

Intelligent Tutoring Systems

Computer Science

Psychology

Education

Human Computer Interfaces
User Modeling

Educational Psychology
Theories of Learning

Interactive Learning
Distance Education

DFKI, Language Technology

WS 2016/17
# Important issues related to the development of ITS

<table>
<thead>
<tr>
<th>1970s</th>
<th>1980s</th>
<th>1990s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem generation</td>
<td>Model tracing</td>
<td>Learner control</td>
</tr>
<tr>
<td>Simple student modeling</td>
<td>More buggy-based systems</td>
<td>Individual vs. collaborative</td>
</tr>
<tr>
<td>Knowledge representation</td>
<td>Case-based reasoning</td>
<td>Situated learning vs. information processing</td>
</tr>
<tr>
<td>Socratic tutoring</td>
<td>Discovery worlds</td>
<td></td>
</tr>
<tr>
<td>Skills and strategic knowledge</td>
<td>Progression of mental models</td>
<td>Virtual reality</td>
</tr>
<tr>
<td>Reactive learning environment</td>
<td>Simulation</td>
<td></td>
</tr>
<tr>
<td>Buggy library</td>
<td>Natural language processing</td>
<td></td>
</tr>
<tr>
<td>Expert systems and tutors</td>
<td>Authoring systems</td>
<td></td>
</tr>
<tr>
<td>Overlay models/ genetic graphs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DFKI, Language Technology

WS 2016/17
Components of an ITS

Domain module

Student module

Tutoring module

Communication module
Knowledge relevant for an ITS (1)

Domain knowledge

Model of expert knowledge

Topics, subtopics, definitions or processes

Skills needed to generate algebra equations, administer medications, …

Student knowledge

Describes how tutor reasons about a student's presumed knowledge

 Represents each student's mastery of the domain

(acquired skills, time spent on problems, hints requested, possible misconceptions, correct answers, preferred learning style)
Knowledge relevant for an ITS (2)

Tutoring knowledge

Teaching strategies: methods for providing remediation, examples, …

Reasoning about the use of materials, feedback, and testing

(empirical observations, learning theories, technology-enabled)

Communication knowledge

Methods for communication – graphical interface, animated agents, dialog

Communication motivates and supports students

Ensures that a tutor follows a student's reasoning
Categories of domains

Well-structured to ill-structured domains

Problem solving domains (e.g., mathematics problems; Newtonian mechanics)
live at the simple and most well-structured end of the axes.
Some simple diagnostic cases with explicit correct answers also exist here
(e.g., identifying a fault in an electrical board).

Analytic and unverifiable domains (e.g., ethics and law)
in which no absolute measurement or right/wrong answers exist
live in the middle of these two axes.
New regions of physics (e.g., astrophysics) also live here
as empirical verification is untenable.

Design domains (e.g., architecture and music composition)
live at the most complex and ill-structured end of the axes.
In these domains novelty and creativity are the goals, not problem solution.
Student models (1)

Task to be accomplished

Observes student behavior and creates a qualitative representation of
cognitive and affective knowledge

Partially accounts for student performance

(time on problems, observed errors), as well as
the student's emotional state and
reasons about how to adjust feedback to
each student’s specific learning needs

Continuously updates the model through the course of the interaction
Student models (2)

**Purpose**

Supply data to the other tutor modules, particularly the *teaching module*,
which adapts the tutor’s responses to student behavior.

Influences choice of problems presented, feedback adaptation, reaction category

Factors include spatial capabilities, gender, media preferences, …

**Motivation**

Customized feedback pivotal for producing efficient student learning

Instruction tailored to the student’s preferred learning approach increases

student *interest* in learning and supports

the *amount* of learning accomplished
Model concepts

Domain model
Experts knowledge (facts, procedures, …), student model part too

Overlay model
Overlay/ proper subset of the domain model (mastery level, missing knowledge)
Problems to represent misconceptions, bug libraries notoriously incomplete

Bandwith
Describes amount and quality of student input available to the student model
Detailed analysis and comparison to expert solutions

Open model
Student can inspect the user model and make corrections/changes
Has the potential as an aid to reflective learning
Cognitive science techniques

Model-tracing tutors

Model of the domain used to interpret student actions and solution paths
Cognitive representation of tasks, mostly a result of careful task analysis
Tutor traces a student's implicit execution of the encoded rules
Assumes that all problem solving steps can be identified and explicitly coded
Assumes that student performs the same reasoning as encoded in the rules

Constraint-based tutors

Pedagogically significant states expressed as constraints
Constraints represent the application of a piece of declarative knowledge
Detect and correct student errors which appear as violated constraints
Constraints represent states the student should satisfy, not the paths involved
Applied in intractable domains, domains that cannot be fully articulated
Model tracing

Method
Problem solving/student behavior modeled by a production system
Rules model “curriculum chunks” of cognitive skills
Production system encompasses space of suitable problem solving ways
(reasoning in physics and mathematics, logical inferencing)

Advantages
Enables monitoring the acquisition of chunks of knowledge (tracing)
Enables feedback on very precise level (individual production rules)
Adaptable to student behavior

Problems
Letting the student explore the consequences of misconceptions
Finding out about failure is associated with big learning gains
Only applicable to modeling procedural skill acquisition
Constraint-based tutoring

Method
Problem solutions checked against a set of constraints
Constraints model domain properties
(syntactic, semantic constraints)
Applicable to “ill-structured” domains where model tracing is infeasible

Advantages
Enables monitoring the acquisition of domain properties
Enables feedback on very precise level (constraint violation)
Caters for variations in solution specifications

Problems
Only applicable to complete solution specifications
Little support for incremental development
No reference to proper problem solving process
Teaching strategies

Didactic strategy
Student's problem solving accompanied by piece-wise explanations
Student in some sense navigates in a normative problem-solving space

Socratic strategy
Student's problem solving is guided and supported by hints
Aims at enabling a student's knowledge construction

Socratic teaching generally considered superior:
Long term benefits achieved, harder for students (takes more time)
Support knowledge transfer (applying skills to related tasks and domains)
Means of communication

*Formal*

- Comparably easy to control
- Interpretation of errors may not be so simple
- Applicable to limited set of domains

*Menu-based/pseudo natural language*

- Also easy to control
- Offers prefabricated choices rather than opportunities for free specifications

*Natural language*

- Notoriously difficult to analyze
- Some success, but severe problems with shallow, robust methods
- Increased learning gain through exercising the precise use of expressions
- Increased difficulties with combined natural language/formula expressions
Categories of reactions/feedback

Accept/reject
Basic form of (rudimentary) feedback

Encouragement
Addresses the student's emotional state; important, but not overdoing

Corrections
Some sort of a combined (and more informative) accept/reject

Hinting
Most important contribution to direct the student, rich variety of hints

Prompting
A specific form of hinting, best achievable with speech

Summary
Description of intermediate/final state of problem solving
Communication of tutors – a classification

**Mixed-initiative dialog**

Only few NL tutors support full mixed initiative dialog

**Single-initiative dialog**

Limited discourse context considered

**Directed dialog**

Tutor controls the dialog, prompts students for explicit information, …

**Finessed dialog**

Dialog simulated through menu-based input, logical forms, …
The geometry tutor

Capabilities

- Supports students in the solution of geometry problems
- Exhibits careful and detailed natural language analysis
  - Natural language reference to, e.g., angles in triangles may be tricky

Motivation

- Students are encouraged to write self-explanations
- Cognitive science studies demonstrate that this increases learning effects
- Promoting self-explanations requires the system to
  - understand them
  - facilitate their elaboration through adequate feedback
Two excerpts of sessions with the geometry tutor

S1: Angles are equal.
T1: Are any two angles congruent? What tells you that there are two congruent angles?
S2: Angles in a triangle are equal.
T2: That is a bit imprecise. Are ALL angles in ALL triangles congruent?
S3: Angles in an isosceles triangle are equal.
T3: Correct, except to: are all three angles in an isosceles triangle congruent?
S4: Base angles in an isosceles triangle are equal. (correct)

S5: They are 180
T5: Could ou be more specific? Who are they? And what tells you they are 180?
S6: The angles are 180
T6: You are heading in the right direction, but you need to state this rule more precisely. First, where does the 180 come from? Are all angles 180?
S7: The angles in a triangle are 180.
T7: You are awfully close. But is each angle measure equal to 180? Precisely what is 180?
S8: The sum of the angles in a triangle are 180. (correct)
Issues in the communication of tutors

Dialog strategies

Only few NL tutors support full mixed initiative dialog

NL analysis

Limited discourse context considered

NL generation

Tutor controls the dialog, prompts students for explicit information, …
The role of emotion in tutoring

Recognising the mood of the student

Interpreting sequences of student actions
(some even monitor facial expressions)
Selecting/adapting system actions to boredom, frustration, enthusiasm, …

Teaching environments

Use of animated agents
Examining the effects of various forms of agents

Preventing misuse

Gaming – just clicking to get maximum feedback in minimal time
Cheating – producing absurd or off-the-topic contributions
Pedagogical agents

Properties of autonomous agents
Perceive their environment
Having an internal goal and can process information
Use artificial intelligence techniques to respond rationally in their environment

Properties of pedagogical agents
Represent a special class of software agents
(goes beyond simply looking good)
Life-like and appear to have emotion along with
an understanding of the student’s problems
providing contextualized advice and feedback throughout a learning episode

Motivation
Provide contextualized problem-solving advice
Evidence suggests that intelligent tutors with lifelike characters are
pedagogically effective and have a strong motivating effect on learners
Authoring tools

Purpose
Facilitates the design and development of tutoring systems

Motivation
Best done by domain experts with domain expertise, but no software skills

Ingredients modeled
Task and example specifications, associated with communication specifications

Modeling
Generally oriented on surface appearances (e.g., natural language text portions)

A fundamental trade-off
between depth of formal reasoning and potential for building authoring tools
Evaluating tutoring systems

Techniques
Specifying conditions to be tested
Experiments with a featured group and a control group
Computing the effect of the difference – statistically significant?

Problems
Experiments are expensive – picking most important properties
Capturing system/teaching properties to be tested

Examples
Geometry curriculum with ITS significantly better than traditional form
Linguistically adequate presentations improve performance significantly
ITS as a scientific field

Research community
Annual conferences since 20 years, approx. 200 participants
A dedicated journal, presence in related conferences and journals

Domains of application
Facilitating learning in groups (classroom)
Ono-to-one tutoring in physics, mathematics, programming, formal design …

Success
Significantly improved learning with ITS
Cognitive tutors for algebra and geometry in use in more than 1300 US schools
ITS technology

Relations to other areas of intelligent systems

Logical inference systems (Automated theorem proving, Constraint systems, …)

Optimized towards efficient problem solving

Unusable without considerable transformations and enhancements

Expert systems (including explanation facilities, e.g., Digital Aristotle)

Reasoning foresees needs of afterwards considerations

Insufficient for problem scaffolding and error handling

Consequences on ITS development

Tension between use of reasoning capabilities and usability of authoring tools

Strongly dependent on complexity of the tutoring task and capabilities
The DIALOG project (SFB 378)

Goal

Participating in a flexible natural language tutorial dialog
Empirically investigating dialogs in teaching mathematical proofs

Architecture – modular design

Learning environment – getting acquainted with some lesson material
Mathematical proof assistant – checks appropriateness of student's utterances
Proof manager – maintains representation of constructed proof object
Natural language processing – NL expressions interleaved with formulas
attempts the interpretation of imprecise, ambiguous and faulty utterances
Dialog manager – maintains state of dialog and determines system reaction
including an embedded hinting algorithm
Knowledge resources – domain and pedagogical knowledge (hint taxomony)
WIZARD-OF-OZ EXPERIMENT 1

Goal

Collecting a corpus on tutorial dialogs in the naive set domain
Testing tutorial strategies developed

Experiment phases

Preparation and pre-test on paper
Tutoring session mediated by Wizard-of-Oz tool
Post-test on paper and evaluation questionnaire

Tasks to prove

(1) $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$
(2) $A \cap B \in P((A \cup C) \cap (B \cup C))$
(3) If $K(B) \supseteq A$, then $K(A) \supseteq B$

Experience gained

Socratic strategy not as effective as hoped (long-term effects unexplored)
Distracted by lengthy clarification subdialogs resolving low-level issues
WIZARD-OF-OZ EXPERIMENT 2

Goal
Collecting a corpus on tutorial dialogs about relations (a more advanced topic)
Exploring human hinting strategies in a socratic style

Experiment phases
Getting acquainted with the domain and environment on the computer
Tutoring session mediated by Wizard-of-Oz and editing tools
Evaluation questionnaire

Tasks to prove
(1) \((R \circ S)^{-1} = R^{-1} \circ S^{-1}\)
(2) \((R \cup S) \circ T = (R \circ T) \cup (S \circ T)\) \hspace{1cm} (for relations \(R, S\) and \(T\) over a set \(M\))
(3) \((R \cup S) \circ T = (T^{-1} \circ S^{-1})^{-1} \cup (T^{-1} \circ R^{-1})^{-1}\)

Experience gained
Mistakes of various kinds (see the categories on the next slides)
Tutor reactions addressing errors opportunistically
**INTERPRETATION OF SLOPPY EXPRESSIONS**

1. \( A \cup B \) must be in \( P((A \cup C) \cap (B \cup C)) \), since \( (A \cap B) \cup C \supseteq \) of \( A \cap B \)

2. If \( A \) is a subset of \( C \) and \( B \) a subset of \( C \), then both sets together must also be a subset of \( C \)

   Relations ambiguous between *element* and *subset*, resp. *union* and *intersection*

3. \( K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D)) \), De Morgan Rule 2 applied to both complements

4. \( A \cap B \) on the *left side* is \( \in \) of \( C \cup (A \cap B) \), which is extended only by \( C \)

   Intended referents not mentioned explicitly, scopus preferences apply
   Metonymic interpretations required
**DOMAIN ONTOLOGY**

*Domain knowledge representation*

- Complete logical definitions represented in $\lambda$-calculus
- Inheritance used to percolate shared information, *no* hierarchical organization
- Only proof-relevant knowledge expressed

*Discrepancy to linguistic requirements*

*Discrepancy bridged through intermediate representation*

- Imposing hierarchical organization
- Linking *vague* and *general* terms to domain terms
- Additionally modeling *typographic* features (markers, orderings)
ANALYSIS PHASES

Preprocessing and parsing

Mathematical expressions substituted with default lexical entries

Syntactic parsing and building linguistic meaning representation

Domain and discourse interpretation (using the semantic lexicon)

Symbolic representation built and passed to the proof manager

Consulting the tutoring manager with results obtained
EXAMPLES REVISITED

(1) $A \cup B$ must be in $\mathcal{P}((A \cup C) \cap (B \cup C))$, since $A \cap B \in (A \cap B) \cup C$
   only ELEMENT interpretation is relevant, SUBSET is incorrect

(2) If $A$ is a subset of $C$ and $B$ a subset of $C$, then both sets together must also be a subset of $C$
   only UNION interpretation is relevant, INTERSECTION merely correct

(3) $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$, De Morgan Rule 2 applied to both complements
   only separate rule application possible, not their composition, thus disambiguated

(4) $A \cap B$ on the left side is $\in$ of $C \cup (A \cap B)$, which is extended only by $C$
   judged as incorrect, since argument types clash with ELEMENT relation
HANDLING ERRORS – CORPUS EXAMPLES

Example formula          Error category
(1) $P((A \cup C) \cap (B \cup C)) = PC \cup (A \cap B)$                  3
(2) $(p \cap a) \in P(a \cap b)$                                     2
(3) $(x \in b) \notin A    K(A) \supseteq x$                          2
(4) $P((A \cap B) \cup C) = P(A \cup B) \cup P(C)$                  1
(5) $P(A \cap B) \supseteq (A \cap B)$                               1
(6) if $K(B) \supseteq A$ then $A \notin B$                          2

3: Structural errors (1): Missing space between $P$ and $C$, and enclosing parentheses
2: Type errors (2,3,6): Typographical (2), argument type (3), operator type (6)
1: Logical errors (4,5): Set inclusion for equality (4), membership for subset (5)
ERROR RECOGNITION

Components contributing

Formula analyzer

Defined repertoire of operators and variables, with arity and type restrictions

Proof manager

Tries to find a proof for the assertion, within the defined context

Error categories

*Structural (syntactic) errors* – Formula analyzer cannot build an analysis tree

*Type (semantic) errors* – Formula analyzer reports a type mismatch

*Logical (truth-value) errors* – Proof manager disproves the assertion
ERROR CORRECTION

Components contributing

Formula analyzer
Performs structural modifications to enable building an analysis tree

Formula modifier
Tries to apply cognitively plausible changes to the flawed formula

Formula modifications
Local and cognitively justified changes
Guided by error category and flawed portion of the formula
Searching for modifications that improve the correctness state of the formula