Educational Technologies
WS2006

Authoring Tools - CTAT

Bruce M. McLaren
Senior Researcher, DFKI
Approximate Plan of the Course

- 18.10.2006 Introduction, Project descriptions and ActiveMath Demo
- 25.10.2006 Representation
- 08.11.2006 Learner Modeling
- 15.11.2006 Web technologies and Security
- 22.11.2006 Tutorial Planning and Instructional Design
- 29.11.2006 Media Principles
- 06.12.2006 Interactive Exercises
- 13.12.2006 Diagnosis: Constraint Based
- 20.12.2006 Diagnosis: Model Tracing und Domain Reasoning
- 10.01.2007 Tutorial Dialogues
- 17.01.2007 Authoring tools, CTAT
- 24.01.2007 Action Analysis and Machine Learning techniques
- 31.01.2007 Cognitive Tools (iCMap, Assembly, Learning Log)
- 07.02.2007 Meta-Cognitive Help (The HelpTutor)
- 14.02.2007 Presentation of student projects
Special Thanks ...

Tom Murray

tmurray@cs.umass.edu

http://www.tommurray.us/

(413) 529-2120

Overview of the Lecture

Overview of Authoring Tools
- What exactly is an Intelligent Tutoring System?
- What types of ITSs have been authored?
- What components must be authored?
- What methods have been used to author ITSs?
- What are some of the issues related to authoring tools?

Case Study of a particular Authoring Tool: The Cognitive Tutor Authoring Tools (CTAT)
- Overview of CTAT
- Brief movie on building tutors using CTAT

Summary and Conclusions
What is an Intelligent Tutoring System (ITS), such that one can be “authored?”

- Any computer-based instructional system that separates content (what) from strategy (how)
- Usually makes inferences about what the student “knows”
  - i.e., Contains a model of domain, strategy, and/or student
- Typically have a mixed-initiative approach in which students can ask questions and have more control over their learning
- “Learn by Doing”
ITS Architecture

All components of the ITS architecture can be (and have been!) authored using ITS authoring tools.
Purposes of ITS Authoring Tools
(Roughly in order of importance or emphasis)

- Cost-effective production of ITSs
- Decrease skill threshold for authors
- Help the designer/author articulate knowledge
- Support good design principles (UI, pedagogy)
- Enable rapid prototyping of ITS designs
- Allow more participation of practicing educators in ITS design and evaluation
What kinds of ITSs have been authored?

- Both pedagogy-oriented and performance-oriented ITSs
- Seven Types of ITSs
- Tools constrain ITSs
### How many ITS authoring tools have been built?

<table>
<thead>
<tr>
<th>CATEGORY</th>
<th>PROJECTS/SYSTEMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Curriculum Sequencing and Planning</td>
<td>DOCENT, IDE, ISD Expert, Expert CML</td>
</tr>
<tr>
<td>2 Tutoring Strategies</td>
<td>REDEEM, Eon, GTE, SmartTrainer AT</td>
</tr>
<tr>
<td>3 Device Simulation and Equipment Training</td>
<td>RIDES, DIAG, MITT-Writer, ICAT, SIMQUEST, XAIDA</td>
</tr>
<tr>
<td>4 Domain Expert System</td>
<td>Cognitive Tutor Authoring Tools (CTAT), ASPIRE, Demonstr8, D3 Trainer, SimStudent</td>
</tr>
<tr>
<td>5 Multiple Knowledge Types</td>
<td>CREAM-Tools, DNA, ID-Expert, IRIS, XAIDA, AuthorIT</td>
</tr>
<tr>
<td>6 Special Purpose</td>
<td>IDLE-Tool/IMap, Leap-AT, BioWorld Case Builder</td>
</tr>
<tr>
<td>7 Intelligent/adaptive Hypermedia</td>
<td>ActiveMath, InterBook, MetaLinks, CALAT, GETMAS</td>
</tr>
</tbody>
</table>

Pedagogy-Oriented Systems - Focus on sequence with canned content

Performance-Oriented Systems - Focus on practice and feedback

> 30 projects

Source: Bruce McLaren Educational Technologies WS 2006/07
1. Curriculum Sequencing and Planning

**Systems:** DOCENT, IDE, ISD Expert, Expert CML

► **Overview:**
  ▶ Basic and early historical systems
  ▶ Organize instructional units into a hierarchy of courses, modules, etc. Typically have instructional objectives.
  ▶ Separates content from presentation and sequencing

► **Strengths:**
  ▶ Rules, constraints, or strategies for “intelligently” sequencing content--at the “macro level” (topic level)
  ▶ Sequencing determined dynamically based on student’s performance, lesson objectives, and the relationships between course modules.

► **Weaknesses:**
  ▶ The most basic; minimal functionality
  ▶ Usually low fidelity interfaces, canned content, simple student models
2. Tutoring Strategies

*Systems*: REDEEM, Eon, GTE, Smart Trainer AT

**Overview:**
- Representing diverse teaching strategies
- Similar to Curriculum Sequencing, except more fine-grained strategies

**Strengths:**
- Micro-level and explicit tutoring strategies
  - Instructional primitives for hints, explanations, examples, reviews, feedback...
  - Instruction can have a more dialogue or conversational feel
- Multiple teaching strategies and meta-strategies

**Weaknesses:**
- Often have low fidelity interfaces, canned content, simple student models
The egg from the mother always contains an X chromosome. The father has one X and one Y chromosome in all his cells. When the sperms are made there are two possibilities: a sperm containing an X chromosome or a sperm containing a Y chromosome.

The Royal Navy uses VLF and LF bands for long-range strategic communications for ships and submarines. (NOTE VLF is one-way, shore to ship only due to the large size of aerial.)

VLF and LF are especially important to submarines as the ground wave can penetrate the Earth's surface or water to a depth of several metres. This allows a submarine to remain submerged whilst receiving a message thus reducing the chance of it being detected.
REDEEM Authoring

**Defining your teaching strategies**

**Strategy name:** Practice

**Teaching strategy features:***
- **Teacher choice** vs **Student choice**
- **Lots of teaching** vs **Lots of testing**
- **Start general** vs **Start specific**
- **Questions given** vs **Student chooses**
- **Lots of help** vs **No help**
- **Answers deduced** vs **Answers given**
- **Post interspersed** vs **Post test at end**
- **Pre-Interspersed** vs **Pre-test at start**
- **All computer** vs **Non-computer tasks**

**Question difficulty:**
- Easy
- Medium
- Hard

**Testing styles:**
- Multiple choice
- Fill in the blank
- Matching
- True/False
- Multiple True

**Other strategies:**
- Create strategy...
- Edit other strategy...
- Delete strategy...
REDEEM’s place in the world - Another perspective on ITS Authoring Tools...

Generality

LAT  Diag  RIDES  XAIDA  Eo  REDEEM

Depth of Knowledge

REDEEM  Eo  XAIDA  RIDES  Diag  CTAT

Ease of Use

RIDES  Diag  Eon  XAIDA  REDEEM
3. Device Simulation and Equipment Training

**Overview:**
- Micro-world/simulation-based learning environments
- Focus on equipment/device operation and maintenance procedures

**Strengths:**
- Authoring and tutoring matched to device component identification, operation, and troubleshooting
- Building the simulation is time consuming, but much of the “tutoring” then comes for free
- Typically, a high fidelity interface, matching the domain well

**Weaknesses:**
- Building the simulation is difficult and time consuming!
- Limited instructional strategies
- Limited student modeling
- Focused on procedural skills (but SIMQUEST is an exception, focusing on conceptual skills)

**Systems:** RIDES, DIAG, MITT-Writer, SIMQUEST, XAIDA
Device Simulation Example: RIDES
4. Domain Expert System

*Systems*: CTAT, ASPIRE, Demonstr8, D3 Trainer, SimStudent

**Overview:**
- Runnable models of problem solving expertise (i.e., “expert systems”)
- Compare student performance with that of model

**Strengths:**
- Deep model of expertise
- Buggy and novice rules included

**Weaknesses:**
- Building an expert system is generally very difficult and time consuming
- Limited to procedural and problem solving expertise
- Limited instructional strategies
Domain Expert System Example: Cognitive Tutor Authoring Tools (CTAT)

Web browser with Example-Tracing tutor embedded in HTML

Flash authoring environment with tutor interface shown

Behavior recorder with behavior graph loaded
Domain Expert System Example:

D3 Trainer Medical Tutor

Most of your chosen tests are not relevant.
Domain Expert System Example: ASPIRE

Overview:

- Constraint-Based Modelling

  “If the relevance condition \( R \) is true, then the satisfaction condition \( S \) ought to be true, otherwise something is wrong.”, e.g.,

  - If the current problem is \( \frac{a}{b} + \frac{c}{d} \), and the student’s solution is \( \frac{a+c}{n} \), then it had better be the case that \( n=b=d \).

Strengths:

- Very efficient computationally
- No need for a problem solver
- No need for a bug library
- Neutral with respect to pedagogy

Weaknesses:

- Difficult to define constraints!
- Time consuming
Domain Expert System Example: ASPIRE

Generate syntax constraints
- Generated by analyzing domain ontology and solution structure
- Restrictions specified in ontology translated into constraints

Generate semantic constraints
- Machine learning algorithm that learns from problems and solutions
5. Multiple Knowledge Types

**Overview:**

- “Gagne Hypothesis:” There are different types of knowledge --> Each has its own instructional methods and representational formalism
- Template-like framework for decomposing content into facts, concepts, and procedures

**Strengths:**

- Based on instructional design theory principles
- Instruction matched to knowledge type, e.g.,
  - Facts taught by repetitive practice
  - Concepts taught by analogies and examples

**Weaknesses:**

- Limited to relatively simple facts, concepts, procedures
- Pre-defined tutoring strategies

**Systems:** CREAM-Tools, DNA, ID-Expert, IRIS, XAIDA
6. Special Purpose

**Systems:** IDLE-Tool/IMap, LEAP Authoring Tool

**Overview:**
- Build tutors for a particular type of task or domain
- Leverages the principle that more specific representation and tutoring strategy can better support the student
- Authoring is more “template” like

**Strengths:**
- Can provide strong authoring guidance and constraints
- Fixed design and pedagogical principles can be enforced

**Weaknesses:**
- Each tool limited to a specific type of tutor; thus, may only appeal to a limited authoring audience
- Inflexibility of representation and pedagogy
7. Intelligent/Adaptive Hypermedia

*Systems*: ActiveMath, InterBook, MetaLinks, CALAT

**Overview:**
- Web-based systems (i.e., e-Learning)
- Similar to Category #1 but also deals with Navigation and (dis)orientation issues

**Strengths:**
- Accessibility and UI uniformity benefits associated with the WWW
- Intelligent filtering, sorting, and annotation of hyperlinks
- Potential for making inferences from large numbers of students

**Weaknesses:**
- Limited interactivity and learning environment fidelity (but increasingly **not** the case with systems like ActiveMath)
- Limited student model bandwidth (although this is becoming less of a constraint!)
Intelligent/Adaptive Hypermedia Example: ActiveMath

Define Problem
• Related Competencies
• Difficulty
• Relation to other problems

Build Finite State Machine
• Transitions
• Conditions on Transitions
• Variables for conditions

Test Finite State Machine
Intelligent/Adaptive Hypermedia Example: ActiveMath

Overview:
- Authoring of Problems
- Representation: Finite State Machine
- Use OpenMath representation to test for semantic equivalence

Strengths:

- Finite State Machine very general representation
- Use of Learner Model to adapt presentation of problems
- Separation of problem representation from strategy
- Parameterizing the solution space

Weaknesses:

- Finite State Machine can be very complex to author - Non-computer scientists might not understand!
- Generic representation of tutorial strategies is missing
Authoring the Interface, Domain, Tutoring, and Student Models

- Interface
- Domain Model
  - Curriculum
  - Simulations
  - Expert Systems
- Tutoring Model
- Student Model
1. Authoring the Interface

Not the strength of authoring tools!

- General software products much better at this.
  - Requires knowledge that often isn't on an ITS team
  - Gives freedom to design a bad interface

Systems with built-in interface authoring tools:
- RIDES (below), Eon, SIMQUEST, CTAT
2. Authoring the Domain model

Authoring Curriculum Knowledge and Structures

- Topics/KUs
- Relationships (e.g. prerequisite)
- Knowl. Type (concept, procedure…)
- Objectives
- Importance
- Difficulty
2. Authoring the Domain model

Authoring Simulations of Devices and Phenomena
2. Authoring the Domain model

Authoring Domain Expertise

D3 Trainer
3. Authoring the Tutoring Model

Vast majority have fixed tutoring strategy …
4. Authoring the Student Model

Interestingly, very few AI approaches, such as Bayesian networks ... (but ActiveMath does use such an approach)
What Authoring/Knowledge Acquisition Methods Have Been Used?

In general, these methods could be used for any of the main components and for any of the seven categories.

1. Scaffolding knowledge articulation with models
2. Embedded knowledge and default knowledge
3. Knowledge management
4. Knowledge visualization
5. Knowledge elicitation and work flow management
6. Knowledge and design validation
7. Knowledge re-use
8. Automated knowledge creation
3. Knowledge Management

- ITSs are elaborate systems and authoring them involves managing a large amount of complex information.

- ITSs are particularly difficult to author because of the many diverse and interconnected types of information they contain.
  - Separation of content and tutoring strategy; but not completely independent!
  - Structure of student model depends on domain model
  - Form of tutoring strategies depend on domain model

- Especially useful tools: Object browsers, Version systems.
3. Knowledge Management

- Topics/KUs
- Lesson Objectives
- Interface objects & screens
- Exercises, examples, pictures
- Teaching Strategy actions
4. Knowledge Visualization

Perhaps the most powerful way to help authors understand and comprehend interconnected knowledge.

Topic or curriculum network tools are the most common knowledge visualization tools in ITS authoring.

- Little so far to help with tutoring strategies.
- REDEEM uses sliders but this is a “low-tech” visualization.
4. Knowledge visualization

LAT - Visualize “conversational grammars”

Leap-AT
4. Automated Knowledge Creation

- Infer or create new knowledge or information, saving the author from having to derive, articulate, or enter this information.

- Example-based programming: RIDES, Demonstr8, SimStudent.

- Generation of new problems and solutions from general principles or rules.
8. Automated Knowledge Creation

Machine learning agent that …

▶ Observes model solutions
▶ Learns problem-solving steps
▶ Outputs a set of production rules

Demonstrate a solution

Production Rules

Rule simplify-LHS:

IF

is-equation( Eq ),
is-lhs( Eq, Lhs ),
polynomial( Lhs ),
all-var-terms( Lhs )

Then

simplify( Lhs, S-lhs ),
enter( S-lhs )
How Are Authoring Systems Designed?

Design Tradeoffs & Open Issues

- The space of design tradeoffs
- General vs. special purpose authoring systems
- Who are the authors?
- Who should author ITS instructional strategies?
Impossible to incorporate all aspects
- prohibitive cost & complexity
- conflicting requirements - Which?
Illustrates why different categories and components have different emphases!

[The design space has 24 (6x4) independent dimensions or axes.]
General vs. special-purpose authoring systems

- One of the most active areas of disagreement!

- e.g., special purpose systems: Leap-AT (only for customer service response) and IDLE-Tool (only for “investigate and decide” learning)
  - Advantage: Greater usability, fidelity, depth
  - Disadvantage: Only for design goals that match the tools
  - Does the demand for a particular type of ITS balance the inflexibility?

- “Training” vs. “Education”
  - Real-world tasks
  - Abstract, higher-order thinking tasks
  - Probably easier to provide support for training, e.g., Leap-AT vs. IDLE-Tool
Who are the authors?

What level of skill & training should be expected?

- Widely varying skill sets:
  - instructional design and instructional theory,
  - classroom pragmatics,
  - graphics/UI,
  - domain knowledge,
  - knowledge engineering,
  - script-level programming...

- IDLE, XAIDA, REDEEM, CTAT: try to allow authoring by teachers and “off the street” domain experts with minimal training
Who should specify/author ITS instructional strategies?

<table>
<thead>
<tr>
<th></th>
<th>PROS</th>
<th>CONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teachers</td>
<td>Practical experience</td>
<td>Not good at articulating or abstracting expertise</td>
</tr>
<tr>
<td>PRACTICAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instructional Designers</td>
<td>Theories are widely used in some circles</td>
<td>Limited to basic knowledge types that are easily represented</td>
</tr>
<tr>
<td>ANALYTIC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Psychologists</td>
<td>Know “how the mind works”</td>
<td>Use 'first principles'—only useful for simple knowledge structures</td>
</tr>
<tr>
<td>THEORETICAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Educational researchers</td>
<td>Empirical studies of tutoring and classrooms</td>
<td>After many years still don't agree on much</td>
</tr>
<tr>
<td>EMPIRICAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer scientists</td>
<td>...end up building the systems…</td>
<td>“Isn’t it just all common sense?”…</td>
</tr>
<tr>
<td>(ACTUAL?!?)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain Experts</td>
<td>Experts just show how they do a task &amp; authoring tool infers the instructional methods</td>
<td>Fixed instructional method</td>
</tr>
<tr>
<td>(i.e. NO acquisition of</td>
<td></td>
<td></td>
</tr>
<tr>
<td>instructional knowledge)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Case Study: A Domain Expert System Authoring Tool

From Carnegie Mellon

Thanks to:
Ken Koedinger, Vincent Aleven, Bruce McLaren, Jonathan Sewall, John LaPlante, Brett Leber, Sandy Demi, Chang-Hsin Chang, Mike Schneider, and many undergraduates ...
But First, some background on Cognitive Tutors ...

ACT-R: A Cognitive Theory of Learning and Performance

Big theory ... key tenets:

- Learning by doing, not by listening or watching

Production rules represent performance knowledge:

These units are: Instruction implications:

modular isolate skills, concepts, strategies
context specific address "when" as well as "how"

Cognitive Tutor Technology: Use ACT-R theory to individualize instruction

Cognitive Model: A system that can solve problems in the various ways students can

Strategy 1: IF the goal is to solve $a(bx+c) = d$
THEN rewrite this as $abx + ac = d$

Strategy 2: IF the goal is to solve $a(bx+c) = d$
THEN rewrite this as $bx + c = d/a$

Misconception: IF the goal is to solve $a(bx+c) = d$
THEN rewrite this as $abx + c = d$
**Cognitive Tutor Technology:**
Use ACT-R theory to individualize instruction

**Cognitive Model:** A system that can solve problems in the various ways students can

If goal is solve $a(bx+c) = d$
Then rewrite as $abx + ac = d$

If goal is solve $a(bx+c) = d$
Then rewrite as $abx + c = d$

If goal is solve $a(bx+c) = d$
Then rewrite as $bx+c = d/a$

$3(2x - 5) = 9$

$6x - 15 = 9$

$2x - 5 = 3$

$6x - 5 = 9$

**Model Tracing:** Follows student through their individual approach to a problem -> context-sensitive instruction
Cognitive Tutor Technology: Use ACT-R theory to individualize instruction

Cognitive Model: A system that can solve problems in the various ways students can

If goal is solve $a(bx+c) = d$
Then rewrite as $abx + ac = d$

Hint message: “Distribute $a$ across the parentheses.”
Known? = 85% chance

If goal is solve $a(bx+c) = d$
Then rewrite as $abx + c = d$

Known? = 45%

Model Tracing: Follows student through their individual approach to a problem -> context-sensitive instruction

Knowledge Tracing: Assesses student's knowledge growth -> individualized activity selection and pacing
Cognitive Tutor Algebra

Course

► Integrated tutor, text, and teacher training
► In computer lab 2 days/week, classroom 3 days/week
► Learn by doing:
   ▶ Project-based
   ▶ Student-centered
   ▶ Cooperative learning
   ▶ Teacher as facilitator
Replicated Field Studies

- Full year classroom experiments
- Replicated over 3 years in urban schools
- In Pittsburgh & Milwaukee

Results:
- 50-100% better on problem solving & representation use.
- 15-25% better on standardized tests.

CTAT motivation: Make tutor development easier and faster!

- Development costs of instructional technology are, in general, quite high
  - E.g., ~300 dev hours per hour of instruction for Computer Aided Instruction (Murray, 1999)
  - Cognitive Tutors:
    - Large student learning gains as a result of detailed cognitive modeling
    - ~200 dev hours per hour of instruction (Koedinger et al, 1997)
    - Requires PhD level cog scientists and AI programmers
  - Solution: Easy to use Cognitive Tutor Authoring Tools (CTAT)


How to reduce the Authoring Cost?

- Less programming, more automation
  - Drag & drop interface construction
  - Demonstration-based programming

- Human-Computer Interaction methods
  - User studies, summer schools, informal & formal comparison studies

- Exploit tools already in use
  - Component-based architecture & standard inter-process communication protocols
Authoring with CTAT

- **Cognitive Tutors**
  - Difficult to build; for programmers
  - General for a class of problems

- **Example-Tracing Tutors**
  - Much easier to build; for non-programmers
  - Limited to a single problem instance

- **Typical Approach**
  - Build Example-Tracing Tutor first
    - Cognitive Task Analysis
    - Rapid prototypes - perhaps sufficient for the problem
  - Develop Cognitive Tutor from Example-Tracing Tutors
    - Examples guide planning
    - Serve as semi-automated test cases
CTAT’s Modular Architecture

**Student Interface**
(or external problem-solving environment)
- Cool Modes
- CyclePad
- Java Swing
- Flash

**Behavior Recorder**

**Learner Management System**
- LMS
- Tutor Shop

**Data Shop**

**Tutor Engine**
- Tertle (Lisp-based) + Model Tracer
- Jess (Java-based) + Model Tracer
- Example Tracer (Java-based)
- Example Tracer (Flash-based)

**Editor**
- Eclipse

**GUI Builder**
- IntelliJ
- Dreamweaver
- Code Warrior
- Netbeans
- Flash MX 2004

**Cognitive Model Development Tools**
- TDK (Lisp-based)
- Jess tools (Java-based)
CTAT’s Track Record

- CTAT-based tutors have been used in experiments in Geometry, Chemistry, Chinese, and French
- Over 200 users
- Four releases in past year (1.4, 1.5, 1.6, 2.0)
- 13 papers published during 2005-2006
- CTAT Web site
  - 36,000 unique visitors in 2005
  - 8,800 through mid 2006
- CTAT Downloads
  - Approximately 1700 through mid 2006
Cognitive Tutors have been successful in raising students' math test scores in high school and middle-school classrooms, but their development has traditionally required considerable time and expertise. With the Cognitive Tutor Authoring Tools (CTAT), creating Cognitive Tutors is both easier for experts and possible for novices in cognitive science. The tools draw on ideas of programming by demonstration, structured editing, and others.

**Project News**

**Download new papers about CTAT** May 5, 2006

Two recently accepted conference papers are now available from the CTAT publications page. The first, to appear in the proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS 2006), reports on the results of a small-scale study that examined efficiency gains in tutor development with CTAT. The second, to appear in the proceedings of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006), details a number of recent CTAT innovations through case studies. Both papers provide overviews of the current state of the CTAT project.

**Milestones in the History of Cognitive Tutors** April 12, 2006

Explore two decades of research and development history on Cognitive Tutors. The timeline also provides audio clips and links to research papers.

**CTAT 1.5.1 Released** December 12, 2005

CTAT 1.5.1 is an interim release that fixes two bugs for unordered groups and advanced student input matching.
Stoichiometry Tutor

Suppose the WHO recommended limit for arsenic in drinking water is equal to 0.000014 grams of arsenite (AsO2-) / L solution. To determine the concentration of arsenite in a solution sample that is safe, one needs to check it against the WHO recommendation. How many grams of arsenite (AsO2-) / L solution are in a sample with 0.58 moles of arsenite (AsO2-) in 100 kiloliters (100 KL) of solution? The result should have 2 significant figures. (Hint: the molecular weight of arsenite (AsO2-) is 106.8 g AsO2- / mol AsO2-.)

**Problem Statement**

**Problem**

<table>
<thead>
<tr>
<th>#</th>
<th>Units</th>
<th>Substance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.58</td>
<td>mol</td>
<td>AsO2-</td>
</tr>
<tr>
<td>100</td>
<td>kl</td>
<td>solution</td>
</tr>
</tbody>
</table>

**Reason**

- Given Value
- Unit Conversion

**Result**

<table>
<thead>
<tr>
<th>#</th>
<th>Units</th>
<th>Substance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.06</td>
<td>.Authentication failed</td>
<td></td>
</tr>
</tbody>
</table>

**Hint**: The goal is to convert the amount of substance in moles to grams by using molecular weight.
French Language Tutor

Jones et al.

Drag each sentence vertically into the correct position. Click on the Hint button if you need guidance.

Click on the Hint button if you need guidance.

Je vais bien, merci. Et toi?
Salut, François!
Comment vas-tu?
Salut, Isabelle!
Pas mal.

Check

One or more sentences are out of place. Use Hints for additional help.
In the lac operon in E. coli, the LacZ gene codes for the catabolic enzyme beta-galactosidase, which metabolizes lactose. The LacI gene codes for the regulatory protein LacI that binds to the O site, a DNA sequence upstream from the LacZ gene, and prevents transcription of the LacZ gene.

When lactose is present, it associates with LacI to form a complex that cannot bind to the O site. While regulation acts on mRNA synthesis, it is assayed as enzyme activity.

For this problem you have two main tasks:
1. Summarize the regulatory system’s correct behavior, describing how the external effector regulates the internal behavior of the cell, and how this behavior is mediated.
2. Describe how the regulatory system’s behavior is changed and what the implications of these changes are.

For each of the mutations described in the first column use the IF and THEN menus to construct an explanation of the mutation’s impact on beta-galactosidase synthesis.

Finally, in the last two columns, indicate if cells having the specific genotypes or mutations will have beta-galactosidase activity when the cells are grown in medium with or without lactose. Use (+) and (−) to indicate high and negligible levels of beta-galactosidase activity.

1. Summarize this regulatory system:
   In this [negative] regulatory system the [repressor protein] LacI prevents the synthesis of [beta-galactosidase].
   The effector lactose induces the synthesis of [beta-galactosidase] by combining with the [LacI] protein to form a complex that cannot bind to the operator (O site).

2. Explain the behavior of the genotypes below:
   Menus for explaining mutation’s impact on protein synthesis
<table>
<thead>
<tr>
<th>Genotypes</th>
<th>If...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>lacI+ lacI+ lacI0+</td>
<td>When a negatively regulated system functions correctly</td>
<td>mRNA transcription is OFF when the regulator is bound.</td>
</tr>
<tr>
<td>lacI+ lacI0 lacI0+</td>
<td>When a repressor does not function</td>
<td>the repressor cannot block transcription.</td>
</tr>
<tr>
<td>lacI+ lacI+ lacI0+ / lacI0</td>
<td>When 2nd functional copy of defective gene/site is present</td>
<td>function is restored if wild type's diffusible product is dominant.</td>
</tr>
<tr>
<td>lacI+ lacI+ lacI0+ / lacI0</td>
<td>When the operator is not bound by the repressor</td>
<td></td>
</tr>
<tr>
<td>lacI+ lacI+ lacI0+ / lacI0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lacI+ lacI- lacI0+</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>lacI+ lacI+ lacI0+ / lacI-</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>lacI+ lacI0 lacI0+ / lacI-</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>

"lacI-s is a mutation that causes the lacI protein to bind more strongly."
Development time comparison

Past estimates of ITS development
200:1 for *robust beta* versions used in real classrooms

Example-Tracing Tutor Development
25:1 for *initial alpha* versions: (Koedinger et al, 2004)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Economics</td>
<td>11</td>
<td>3600</td>
<td>2190</td>
<td>180</td>
</tr>
<tr>
<td>Math Assistments</td>
<td>20</td>
<td>810</td>
<td>1170</td>
<td>98</td>
</tr>
<tr>
<td>LSAT</td>
<td>3</td>
<td>240</td>
<td>3000</td>
<td>180</td>
</tr>
<tr>
<td>Language Learning</td>
<td>8</td>
<td>210</td>
<td>575</td>
<td>50</td>
</tr>
<tr>
<td>Totals</td>
<td>4860</td>
<td>6935</td>
<td>508</td>
<td>23.2</td>
</tr>
</tbody>
</table>

If creating a robust beta doubles dev time, we would have ~50:1 ratio -- a reduction of 4 times!

CTAT Example-Tracing Tutor
Web-Delivery Options

- Java
  - WebStart
  - Utility may depend on the situation
  - Still gaining experience with this; can be a bit tricky

- Flash (from Macromedia)
  - With plug-in, compatible with virtually all browsers
  - Easy to deploy
Authoring an Example-Tracing Tutor

Step 1: Create a User Interface
- Create the graphical user interface (GUI) used by the student

Step 2: Demonstrate Behavior
- Demonstrate correct, alternative correct, and incorrect solutions

Step 3: Generalize
- Specify how demonstrated behavior generalizes within given problem
  - allowed order of steps
  - allowed variants for a given step

Step 4: Annotate the Graph
- Annotate solutions steps in the resulting “behavior graph” with:
  - hint messages,
  - error messages,
  - labels for concepts or skills associated with actions

Test and Iterate on Steps 1-4 …

Step 5: Publish Tutor to Web
Movie Showing How an Example-Tracing Tutor is built
## Authoring Tool Use

<table>
<thead>
<tr>
<th>Category</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Early prototypes and proofs of concept</td>
<td>D3 Trainer, Demonstr8, DIAG, IRIS, Expert-CML, SmartTrainer AT</td>
</tr>
<tr>
<td>2. Evaluated or used prototypes</td>
<td>CREAM-Tools, DNA, Eon, GTE, IDLE-Tool, LAT, SimStudent</td>
</tr>
<tr>
<td>3. Moderately evaluated or used</td>
<td>REDEEM, SIMQUEST, XAIDA, ActiveMath, ASPIRE</td>
</tr>
<tr>
<td>4. Heavily used (relatively)</td>
<td>IDE, CALAT, RIDES, CTAT</td>
</tr>
</tbody>
</table>

**Examples:**
- XAIDA domains: equipment operation and maintenance, algebra, medicine, computer literacy, biology
- IDLE-Tool: three informal trials with 21, 8, 8 grad student and grade school teacher authors
Summary

Many types of ITSs have been “authored”

Wide variety of knowledge acquisition and authoring methods have been used

ITS Authoring Tools have matured substantially in past decade

- Some tools have significant use and a few are in commercial or near-commercial form. But mostly still research vehicles …

Promising results in from evaluations of usability and productivity (although we didn’t discuss this in the talk)

- Three studies: REDEEM showed authoring to tutorial time ratios between 2:1 and 3:1
- CTAT has gotten reduction to ~25:1 for more knowledge intensive ITS development
Conclusions--How Easy Can It Be?

There are limits!

- Limited use of cookie-cutter special-purpose authoring tools -- too restrictive for most authors
- Limited ability to reduce ITS authoring to easy, small, independent steps (recipes)
- Authors need to think about the big picture and need skills and tools to do this
...Back to the Future

Connection to standard software development

- Customizability requirements will usually lead the author to specify BEHAVIORS (choices, rules, algorithms) as well as static information.
- This requires ability to RUN, test, and modify these behaviors.
- This is (simple) PROGRAMMING.
- Debugging skills and tools will be needed! (Tracing, stepping, inspecting states, etc.)

Three major questions to answer

1. How much can the difficult task of modelling be scaffolded?
2. Special-purpose tools: Degree of specificity vs. generality?
3. Will ITSs ever be in enough demand to warrant effort?

Chicken-and-egg problem: Need authoring tools to quickly develop successful ITSs; Need successful ITSs to warrant the need for authoring tools.