Engaging Students by Making Computer Science Concepts Come Alive

Susan H. Rodger
Duke University

November 21, 2014
Indiana University

Supported by NSF Grants DRL-1031351, DUE 1044191, CRA, and IBM Faculty Awards

Engaging Students in CS - Back in 1989

• New Assistant Professor at Rensselaer Polytechnic Institute

Rensselaer

• New Course: Combined automata theory with CS1 and CS2 (data structures)
• Student wanted feedback on all the answers in the book!

Different Types of Learners

• Learning Styles
  – Visual Learners
    • Learn through seeing
    • Learn best from visual displays
  – Auditory Learners
    • Learn through listening
    • Learn best through verbal lectures, discussions
  – Kinesthetic Learners
    • Learn through moving, doing and touching
    • Learn best through hands-on approach

How do you reach all three types?

• You must do all three!
  – Provide pictures, diagrams
  – Discuss what you are doing
  – Provide activities for trying it
Outline

• CS Concepts Come Alive with Software
  • Automata Theory with JFLAP
  • Learning Programming with Alice
  • Algorithm Animation

• Challenges in Designing Educational Software

• CS Concepts Come Alive in other ways
  • Manipulatives
  • Group Activities
  • Edible CS

Learner Engagement Taxonomy with visualization software

• Different forms of Learner engagement
  – No Viewing
  – Viewing
  – Responding
  – Changing
  – Constructing
  – Presenting

• ITiCSE Working Group Report 2002 (Naps et al.)

Outline

• CS Concepts Come Alive with Software
  – Automata Theory with JFLAP
  – Learning Programming with Alice
  – Algorithm Animation

• Challenges in Designing Educational Software

• CS Concepts Come Alive in other ways
  • Manipulatives
  • Group Activities
  • Edible CS

Automata Theory

• Foundations of Computer Science (theory)
• Formal Languages
  • Finite state machines, Turing machines, grammars

• Proofs

• Motivation
  • How does a compiler work?
  • How does it recognize the syntax errors in your program?
Formal Languages and Automata Theory

- Traditionally taught
  - Pencil and paper exercises
  - No immediate feedback
- More mathematical than most CS courses
- Less hands-on than most CS courses
- No programming? Unlike most other CS courses

Why Develop Tools for Automata?

<table>
<thead>
<tr>
<th>Textual</th>
<th>({q_0, q_1, q_2}, {a, b}, \delta, q_0, {q_2})</th>
</tr>
</thead>
</table>
| Tabular  | \(\begin{array}{c|c}
            a & b \\
            \hline
            q_0 & q_1 \\
            q_1 & q_2 \\
            q_2 & q_1 \\
          \end{array} \) |
| Visual   | ![Automaton Diagram] |
| Interactive | ![Interactive Diagram] |

Overview of JFLAP

- Java Formal Languages and Automata Package
- Instructional tool to learn concepts of Formal Languages and Automata Theory
- Topics:
  - Regular Languages
  - Context-Free Languages
  - Recursively Enumerable Languages
  - Lsystems
- With JFLAP your creations come to life!

Thanks to Students - Worked on JFLAP and Automata Theory Tools

- NPDA - 1990, C++, Dan Caugherty Over 20 years!
- JFLAP - 1996-1999, Java version
  Eric Gramond, Ted Hung, Magda and Octavian Procopiuc
- Pâté, JeLLRap, Lsys
  Anna Bilska, Jason Salemme, Lenore Ramm, Alex Karweit, Robyn Geer
- JFLAP 4.0 – 2003, Thomas Finley, Ryan Cavalcante
- JFLAP 6.0 – 2005-2008 Stephen Reading, Bart Bressler, Jinghui Lim, Chris Morgan, Jason Lee
- JFLAP 7.0 - 2009 Henry Qin, Jonathan Su
- JFLAP 8.0? – 2011-14 Julian Genkins, Ian McMahon, Peggy Li, Lawrence Lin, John Godbey
DFA Example

• Build a deterministic finite automaton (DFA) to recognize binary numbers with an even number of 1s that are an even number.
• Only use symbols 0 and 1
• Binary numbers: 0, 1, 10, 11, 100, 101, 110, 111, ...
• When is a binary number an even number?
  – Ends in 0
• Which strings should be accepted?
• 11010, 10010, 1111, 10100

No, odd    Yes    No, ends    Yes
no. of 1’s  1

Simulation on 1101010
Accepts Input! 1101010

Build with JFLAP

Simulation on 1101010
Accepts Input! 1101010
Add meaning to states!

Test Multiple Inputs

Another Example: Grammar

- Grammar – set of replacement rules to define a language
- Examples:
  - Grammar for English
    - defines English sentences
  - Grammar for Python programming language
    - defines syntactically correct programs
  - Grammar for a formal language (simpler)
Grammar for $a^n b^n c^n$

- Unrestricted grammar
- Generates strings with an equal number of a’s, b’s, c’s
- a’s first, then b’s, then c’s
- Example strings can derive:
  - abc
  - aabbcc
  - aaabbbccc
  - aaaaabbbbbcccccc
  ...
Example Derivation for aabbcc

\[
S \rightarrow AX \\
\rightarrow aAbcX \\
\rightarrow aaBBbcbcX \\
\rightarrow aabBbcbcX \\
\rightarrow aabbDcX \\
\rightarrow aabbDCX
\]

rule: \( S \rightarrow AX \)
rule: \( A \rightarrow aAbc \)
rule: \( A \rightarrow aBbc \)
rule: \( Bb \rightarrow bB \)
rule: \( Bc \rightarrow D \)
rule: \( Db \rightarrow bD \)

Note: the D absorbed the c!
Example Derivation for aabbcc

\[
S \rightarrow AX \\
\rightarrow aAbcX \\
\rightarrow aAbbcX \\
\rightarrow aabBbcX \\
\rightarrow aabDbcX \\
\rightarrow aabbDcX \\
\rightarrow aabbcDX \\
\rightarrow aabbcEXc
\]

Eventually ... \rightarrow aabbcc

We could have done this derivation of aabbcc with JFLAP.

Now let’s see how JFLAP visualizes this derivation with a “parse tree”
Note all letters there, but wrong order: aabcbc

Absorb the “c”

Spit out the “c” at the right end

Absorb second “c”
Spit the “c” out at right end

What else can JFLAP do?

- Create other machines
  - Moore and Mealy
  - Pushdown Automaton
  - Turing machine

- Parsing of grammars
  - regular, context-free grammars
  - Unrestricted grammar

- Conversions for proofs
  - NFA to DFA to minimal DFA
  - NFA \( \leftrightarrow \) regular expression
  - NFA \( \leftrightarrow \) regular grammar
  - CFG \( \leftrightarrow \) NPDA

JFLAP - L-Systems

- L-Systems may be used to model biological systems and create fractals.
- Similar to Chomsky grammars, except all variables are replaced in each derivation step, not just one!
- Commonly, strings from successive derivations are interpreted as strings of render commands and are displayed graphically.
L-Systems
The same stochastic L-system, rendered 3 different times all at the 9th derivation.

Using JFLAP during Lecture
• Use JFLAP to build examples of automata or grammars
• Use JFLAP to demo proofs
• Load a JFLAP example and students work in pairs to determine what it does, or fix it if it is not correct.
JFLAP’s use Outside of Class

• Homework problems
  – Turn in JFLAP files
  – OR turn in on paper, check answers in JFLAP
• Recreate examples from class
• Work additional problems
  – Receive immediate feedback

Two-year JFLAP Study
2005-2007

Fourteen Faculty Adopter Participants

- Duke
- UNC-Chapel Hill
- Emory
- Winston-Salem State University
- United States Naval Academy
- Rensselaer Polytechnic Institute
- UC Davis
- Virginia State University
- Norfolk State University
- University of Houston
- Fayetteville State University
- University of Richmond
- San Jose State University
- Rochester Institute of Technology

Conclusions From Study

• Results of Study showed
  – All the faculty used JFLAP in their courses, mostly for homework, some in lecture
  – Students had a high opinion of JFLAP
  – Majority of students felt access to JFLAP
    • Made learning course concepts easier
    • Made them feel more engaged
    • Made the course more enjoyable
  – Over half the students used JFLAP to study for exams
  – Over half the student thought time and effort using JFLAP helped them get a better grade.

JFLAP is free

www.jflap.org

JFLAP tutorial
Outline

- CS Concepts Come Alive with Software
  - Automata Theory with JFLAP
  - Learning Programming with Alice
  - Algorithm Animation
- Challenges in Designing Educational Software
- CS Concepts Come Alive in other ways
  - Manipulatives
  - Group Activities
  - Edible CS

Alice Programming Language

- Create interactive stories or games
- Learn programming in an easy way, drag-and-drop your code
- Problem solving with visual feedback
  - Objects are visual!
- Alice is free: www.alice.org
- Developed by Randy Pausch

Adventures in Alice Programming
www.cs.duke.edu/csed/alice/aliceInSchools

- 2-week Teacher workshops
  - Over 200 teachers, middle school, high school, some elementary
  - First week Teach Alice, Practice
  - Second week - Develop Lesson Plans
  - All disciplines: math, science, history, language arts, foreign language, art, music, business
  - Summers 2008-2015, funding for lodging
- Main Sites:
  - Duke University, Durham, NC
  - Charleston/Columbia, SC
  - San Jose, CA (started 2014)

Curriculum Materials
www.cs.duke.edu/csed/alice/aliceInSchools

- Over 90 tutorials available for free
- Beginner, advanced, challenges, projects
- Paper handouts and video
- Over 200 Teacher lesson plans
  - Organized by discipline and grade level
Computer Science Concepts come alive with Alice - Examples

- Objects are visible
- Variables
- Inheritance
- Lists
- Array

- Created Challenges – complete the program

Example: Objects are visible

Getting Started Tutorial teaches
- Placing objects
- Moving objects
- Setting up Camera tripods and moving between views
- Using built-in methods and writing your own
  - Dragon flapWings
- Gluing objects together
- Adding sound, 2D pictures to enhance world

Getting Started Tutorial – 3 part

Variables – Timer and Score
Variables – Scores/Timers
Game: Eragon

4 tasks to win the game

Example - Inheritance

- Start with a chicken object
- Rename it to TalentedChicken
  - Change its color
  - Resize it larger
  - Add new methods (jump, fly, scurry)
  - Add events for this chicken
- Save this new class
  TalentedChicken that inherits from the Chicken class

Example - List

Example – Arrays
Shuffle, then Selection Sort

Sort by height
Harry Potter Challenge

• Mix of programming and math challenges

Harry Potter – Math/Computing Level 1 Charms - before

Outline

• CS Concepts Come Alive with Software
  • Automata Theory with JFLAP
  • Learning Programming with Alice
  • Algorithm Animation

• Challenges in Designing Educational Software

• CS Concepts Come Alive in other ways
  • Manipulatives
  • Group Activities
  • Edible CS
Algorithm Animation
Software/Aps/Videos

- AlgoViz.org – collection of algorithm visualizations
- Samba, Jsamba - Stasko (Georgia Tech)
- AnimalScript – Roessling (Darmstadt Univ of Tech, SIGCSE 2001)
- TRAKLA2 – Software Visualization Group – TKK Finland
- Lots of animations and systems on the web!
- Lots of videos of algorithm animations on the web!

Use of Algorithm Animation in CS 1/2

- Instructor
  - Make/Use animations for lecture
  - Stop/Pause – ask what will happen next
  - must be interactive
- Student
  - Create animations
  - Replay animations from lecture with same or new inputs

Lots of other software/programs for algorithm animation

- Red Black Tree – animation on web page

Another red-black tree animation

Student must have graduated. Link no longer works!

http://aleph0.clarku.edu/~achou/cs102/Examples/bst_animation/RedBlackTree-Example.html

Electronic Textbooks (ebooks) engage students

- OpenDSA (Shaffer, Virginia Tech)
  - Algorithm animations built in
- runestoneinteractive.org (Brad Miller, Virginia Tech)
  - Several books (Python)
    - Python - try and run code built in
    - Quizzes
- Zyante.com – interactive textbooks
- Track student progress
- Requirements and design strategies for open source interactive computer science eBooks
  - ITiCSE 2013 Working Group (Korhonen, Naps, et al)

Outline

- CS Concepts Come Alive with Software
  - Automata Theory with JFLAP
  - Learning Programming with Alice
  - Algorithm Animation
- Challenges in Designing Educational Software
- CS Concepts Come Alive in other ways
  - Manipulatives
  - Group Activities
  - Edible CS

Make your tool as interactive as possible – but not too tedious!

- User shouldn’t type everything
- Sometimes select
Allow user to proceed on if they got it

- Complete the rest for them
- Complete parts for them

Avoid Too Many Pop up windows

- OLD JFLAP LR PARSE TOOL

Add Pause/Checkpoint questions

- Allow for pause to think about what comes next
- Undo/go back
- Pop up a quiz question to see if the user understands what he/she just did
  - JHAVE tool does this
  - Can integrate into ebooks

What can make the tool more useable?

- Annotations on states
- Multiple run window
  - Develop test data
  - Easier for grading
- General definitions
  - FA – recognize one or more symbols
  - NPDA – pop or push 0 or more symbols
- Batch processing
Naming your software

What is a “good” name for your tool?

JFLAP name is unique

JAWAA name is not unique

How popular is JAWAA?

Outline

• CS Concepts Come Alive with Software
  – Automata Theory with JFLAP
  – Learning Programming with Alice
  – Algorithm Animation
• Challenges in Designing Educational Software
• CS Concepts Come Alive in other ways
  – Manipulatives
  – Group Activities
  – Edible CS
Interaction in Class – Props
Passing “Parameters” in Class

• Pass by reference – throw frisbee

• Pass by value – throw copy of frisbee

• Pass by const reference – throw “protected” frisbee

Interaction in Class – Props
Linked List and Memory Heaps

ITiCSE 98 – Astrachan – “Concrete Teaching: Hooks and Props as Instructional Technology”

Interaction in Class – Props
Memory Heap

MEMORY HEAP
Example: Be a Robot

- 4 People
  - Controller (head)
  - Sensors (eyes)
  - Manipulators (2 hands)
- Blindfolded except eyes
- Controller knows what to build
- Limited communication

SIGCSE 96, Rodger, Walker

Cards

- Card Class – shuffling, dealing hands
- Poker hands – Full house, Flush, etc.

Notable Women in Computing Cards

bit.ly/NotableW

- Based on Wikipedia project – wrote guide on how to write a Wikipedia page on a Notable women in Computing
- Picked 54 Women - deck of cards
- Page on using cards to teach CS
- Can buy a deck
  - Have decks with me for $6/deck
  - Early register for SIGCSE 2015
  - Kickstarter now til Nov 8

Outline

- CS Concepts Come Alive with Software
  - Automata Theory with JFLAP
  - Learning Programming with Alice
  - Algorithm Animation
- Challenges in Designing Educational Software
- CS Concepts Come Alive in other ways
  - Manipulatives
  - Group Activities
  - Edible CS
Engaging students in a group activities/large course

- Problem Solving in groups
  - Clickers, Google forms – compare results
  - Flip Classroom, reading quizzes (turn off at start of class)
- Acting out stories, games
  - *Everything I needed to know about teaching...* - Pollard, Duvall (SIGCSE 2007)
- Acting out algorithms with the whole class
  - Make a binary tree with the whole class
  - Calculate the height of the tree
  - *Making Lemonade ... large lecture classes* – Wolfman (SIGCSE 2002)
- Acting out algorithms with a subset of students
  - Sorting algorithms – selection sort, insertionsort, etc
  - CS Unplugged activities

### Middle School students sorting themselves with Bubblesort

### Outline

- CS Concepts Come Alive with Software
  - Automata Theory with JFLAP
  - Learning Programming with Alice
  - Algorithm Animation
- Challenges in Designing Educational Software
- CS Concepts Come Alive in other ways
  - Manipulatives
  - Group Activities
  - Edible CS

### About Me - Hobby – Baking Shape cakes

- The Wiggles magazine Issue No. 42
How do you make those cakes?

What happens when your hobby and your career collide?

It is now time for engaging students with edible CS

Automata Theory
Interaction in Class – Props
Edible Turing Machine

- TM for \( f(x) = 2x \) where \( x \) is unary
- TM is not correct, can you fix it? Then eat it!
- States are blueberry muffins

Students building DFA with cookies and icing
CS 2 – Data Structures
Red-Black Tree (cookies)

CS 1
Animated Bouncing Smileys

CS 1
Sorting Cookies

Discrete math
A graph and its Dual Graph
Discrete Math
The Pancake Problem

Alice Programming Language

More Alice Programming Language

Conclusions
• We have shown several ways to engage students with CS concepts and make those concepts come alive
• Questions?