Computing Computational Thinking

Alexander Repenning & David Webb
Computer Science & School of Education
University of Colorado
Computational Thinking Tools
Alexander Repenning

Not the usual suspects: inner city, remote rural, Native American Alaska, California, Wyoming, South Dakota, Colorado, Texas, ...

users

projects

Inflatable Icons
3D simulation authoring for everyone
AgentCubes

AgentSheets
collective simulations
Mr Vetro

CS curriculum from elementary to graduate school

programming for everyone

funding

NSF
National Institutes of Health
EU
Google
**goal**: get computer science in to public schools
- inner city, remote rural, Native American schools

**approach**:  
1. scalable game design to motivate and educate students to learn about computer science through game design ranging from middle to graduate school  
2. start with game design, move on to science simulation building  
3. broaden participation through getting game/simulation design into required courses if possible (neighborhood: keyboarding, powerpoint)

**results**:  
- already > 4000 students (Alaska, Colorado, Ohio, Oregon, South Dakota, Tennessee, Texas, and Wyoming)  
- 45% girls, 63% of girls want to continue  
- some middle schools 600 students/year  
- new game every 14sec during school hours
Computational Thinking definition

Good news: we have come a long way

- **2009**
  - CT ≠ Programming
  - example: “grandma backing a cake”

- **2011: CSTA, ISTE, NSF:** Computational thinking (CT) is a problem-solving process that includes (but is not limited to) the following characteristics:
  - **Formulating problems** in a way that enables us to use a computer and other tools to help solve them.
  - Logically **organizing and analyzing data**
  - Representing data through abstractions such as **models** and **simulations**
  - Automating solutions through **algorithmic thinking**
  - Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources
  - **Generalizing** and **transferring** this problem solving process to a wide variety of problems
I want to be able to walk into a classroom with game design and ask a student: “...now that you can make space invaders, can you also make a science simulation?”

– Len Scrogan, Director of Instructional Technology, Boulder Valley School District
the trouble with transfer

- good news: transfer connects well to CSTA, ISTE, NSF operational CT definition
- but:

  This idea--that programming will provide exercise for the highest mental faculties, and that the cognitive development thus assured for programming will generalize or transfer to other content areas in the child's life--is a great hope. Many elegant analyses offer reasons for this hope, although there is an important sense in which the arguments ring like the overzealous prescriptions for studying Latin in Victorian times.

  - Roy Pea, in Logo Programming and Problem Solving, 1987
challenges for transfer 2.0

- need to find realistic expectations
- need to find the right level representations, e.g., computational thinking patterns
  - need to make CT patterns computable
  - need to make CT patterns recognizable by people
the “right” level of representation for transfer

program

phenomena

frogger

Michotte: “Perception of Causality”

launch:

collision, push, pull, diffusion, hill climbing, ...

loop, if, then, else, print, ...

avalanche

import string
if _name_ == _main_
if len(sys.argv) == 2:
print "Gregorian calendar.
else:
print "Julian calendar.
if int(sys.argv[1]):
print "Julian calendar.
else:
print "Gregorian calendar.
if int(sys.argv[1]):
print "Julian calendar.
else:
print "Gregorian calendar."
computational thinking
inventory

game design computational
science

Maslow's hierarchy
cooperative diffusion

Brownian motion
transfer diffusion

Push pull transport

feedback sensing noise

subsumption architecture

robotics
basic computational thinking patterns

- **Collision**: Frogger: Frog meets Truck
- **Push**: Sokoban: person pushes boxes
- **Transport**: Frogger: logs transport frogs
- **Generate**: Space Invaders: defenders shoot rockets
- **Absorb**: Bridge Builder: tunnel absorbs cars
- **Choreography**: Space Invaders: mother ship makes attack alien ships move left and right and descend
- **Polling / Counting**: Pacman: game over when all the dots are eaten
advanced patterns

- **Diffusion**: Electricity, Heat, rumors, toys: spread of information
- **Seeking**: Sims: people finding food
- **Collaborative Diffusion**: Soccer: players collaborate and compete
- **Multiple Needs**: Maslow’s hierarchy of needs
computing computational thinking

Latent Semantic Analysis inspired similarity

user control

hill climbing

generation

absorption

collision

diffusion

pull

drive

push

transport

game/simulation
Computational Thinking
Pattern Analysis Graph

Scalable Game Design Arcade

Probot
Program the robot so it can get through the maze to the flag. The commands that you can use are “forward”, “turn left”, “turn right”, and “change to red” (you need to be red to cross the lava). Use the drawing tool to place the commands into the boxes. Once you have placed the commands into the boxes, press spacebar to have the robot start. R resets everything. To win you must finish the game in 12 move or less, remember, commands on the same line all happen at the same time.

Run

Similarity Score to Four Tutorial Games
This score shows how much your game structure is similar to the tutorial games. Max value is 1

This game’s similarity score to Frogger: 0.624
This game’s similarity score to Sokoban: 0.715
This game’s similarity score to Space Invaders: 0.644
This game’s similarity score to Sims: 0.082

Similarity Score Matrix
Below Matrix shows other AgentSheets projects sharing similar programming structure.

This Matrix updates itself every 2 and half hours. It may have random projects right after your submission.
TRANSFER?

“Now that you can make ‘Space Invaders’, can you also program a science simulation?”
ONE STUDENT MADE DIFFERENT GAMES

Game #1
- Sokoban

Game #2
- Sims
Science Simulation based on Chaos theory
Transfer
From Game Design to Science Simulation Design

Game #1 and #2

Science simulation
conclusions

- we can compute computational thinking
- we need to come up with concrete benchmarks of transfer
- there are early indicators for transfer between game design and science simulation design. But need to be careful:
  - do not confuse correlation with causation
  - investigate role of teacher to scaffold concepts to be transferred.
Thank you!

Scalable Game Design

Please join us

http://scalablegamedesign.cs.colorado.edu