Teaching computer science to K-8 students at risk for academic failure: Research findings and implications for practice

Maya Israel

misrael@illinois.edu

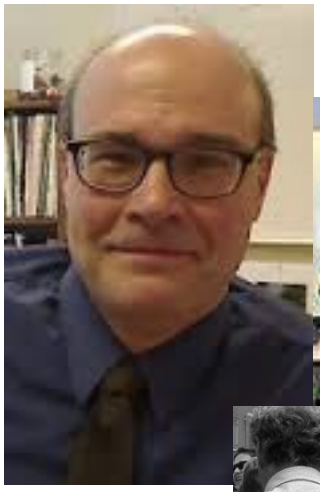
@misrael09

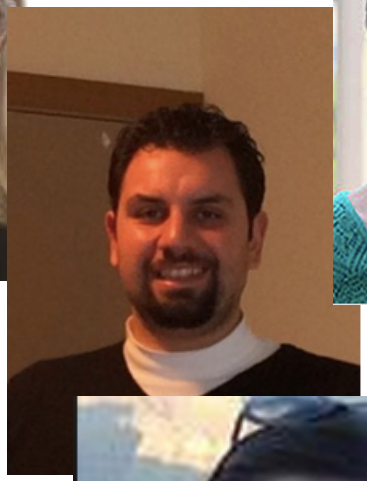
Roadmap

- What is K-12 Computer Science (CS) for All in the U.S. ?
- Challenges of students with disabilities in K-12 CS education.
- Methodologies for:
  - integrated computing into mathematics
  - studying programming with students with disabilities
- Approaches to including students with disabilities in K-12 CS education

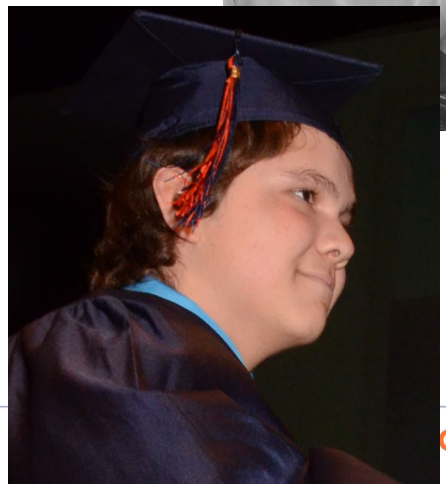
****PLEASE ASK QUESTIONS THROUGHOUT**

First…this is a collaborative effort





CS is beginning to be seen as foundational knowledge for all students as one of the STEM areas

CS is now in the newly reauthorized education act (Every Student Succeeds Act, 2016) as a foundational content area

What is CS for All in the U.S.?

- An educational movement to include CS education opportunities for ALL students.
  - White House Initiative 2016
  - Petition signed by 26 governors and dozens of industry CEOs
  - National Science Foundation investment
  - State and district initiatives

K-12 CS Education

- K-12 opportunities that teach concepts and practices associated with computing from early grades
- Learning to create tech vs. consuming tech
- Why: Disproportionality issues

Bootstrap
+ computing creatively thriving mathematically

CS PRINCIPLES

SCRATCH
forever
imagine
program
share

CODESTERS
Coding in Your Classroom

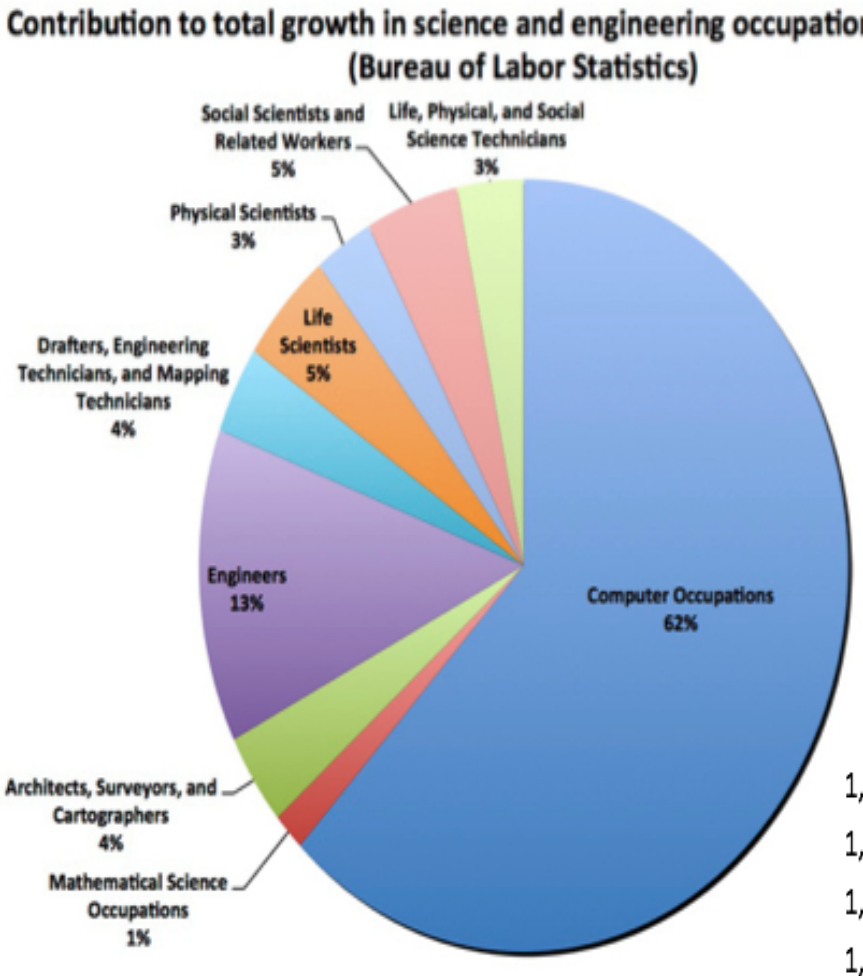
C O
D E

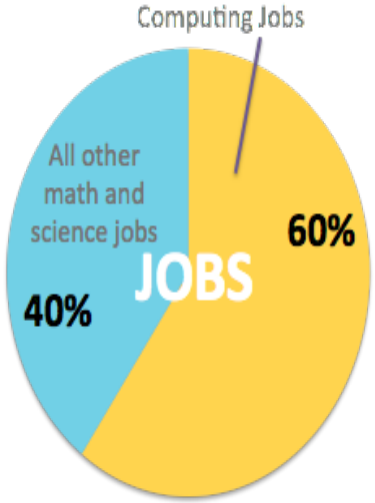
K-12 CS Framework Development

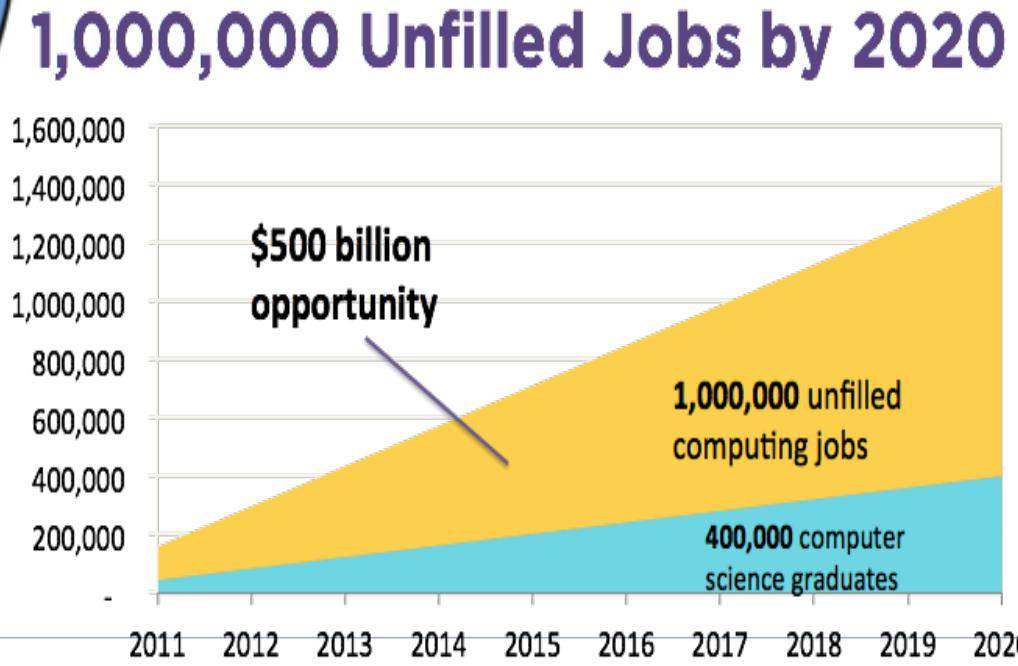
- National initiative to identify the concepts and practices that all students should understand.
- Just finished 3rd public review
- Standards developed from this framework
- https://k12cs.org/

Why focus on computing?

- **Jobs Argument:**
  - US Dept. of Labor Statistics says that by 2020, there will be 1 million computing jobs, but only 30% will be filled at the current rate.
- **Beyond the STEM pipeline argument:**
  - Real-world application of mathematics, opportunities to practice problem solving, persistence, collaboration
  - Equity







From Code.org

Sources: BLS, NSF, Bay Area Council Economic Institute

What about students with learning disabilities?

- Challenges
- Data collection and analysis
- Strategies with promise

What are learning disabilities (LD)?

- "A disorder in one or more of the basic psychological processes involved in understanding or using language, spoken or written, which disorder may manifest itself in the imperfect ability to listen, think, speak, read, write, spell, or do mathematical calculations" (20 U.S.C. §1401(30)).

- Largest category of students receiving special education services in the U.S.--Approximately 2.4 million school children (Cortiella & Horowitz, 2014).

IDEA description of Disability

- "Disability is a natural part of the human experience and in no way diminishes the right of individuals to participate in or contribute to society. Improving the educational results of children with disabilities is an essential element of our national policy of ensuring equality of opportunity, full participation, independent living, and economic self-sufficiency for individuals with disabilities"
  - [IDEA, 20 U.S.C. Sec. 1400(c)(1)]

Challenges of Students with Learning Disabilities in CS

- Inaccessible technology & curricula
  - Students with print-based LD struggle with text-based programming languages.
  - LD influences memory and causes difficulty with multi-step complex problem solving.

Ladner & Israel (in press)

Challenges (cont.)

- CS is meant to be fun, creative, and exploratory.

BUT…..this type of exploration is challenging without background knowledge and skills.

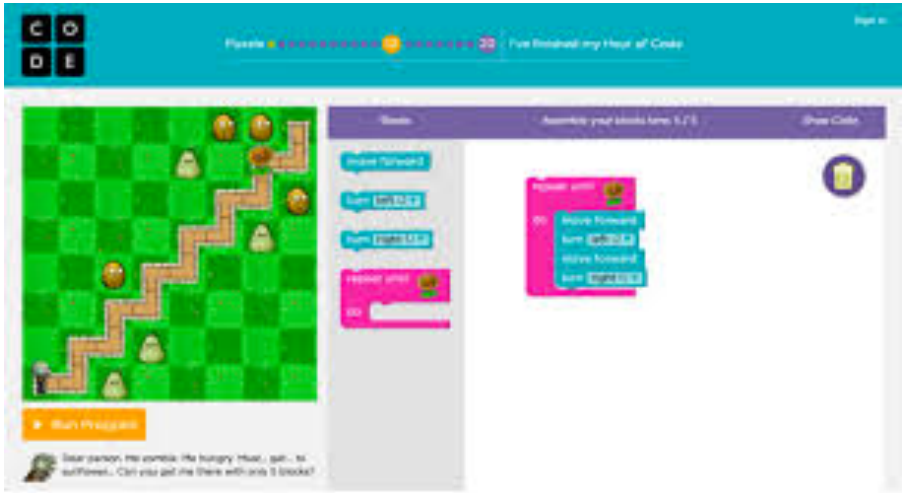
- So....many of these learners quit when they reach a difficult computing problem.

Our Research to Practice Approach

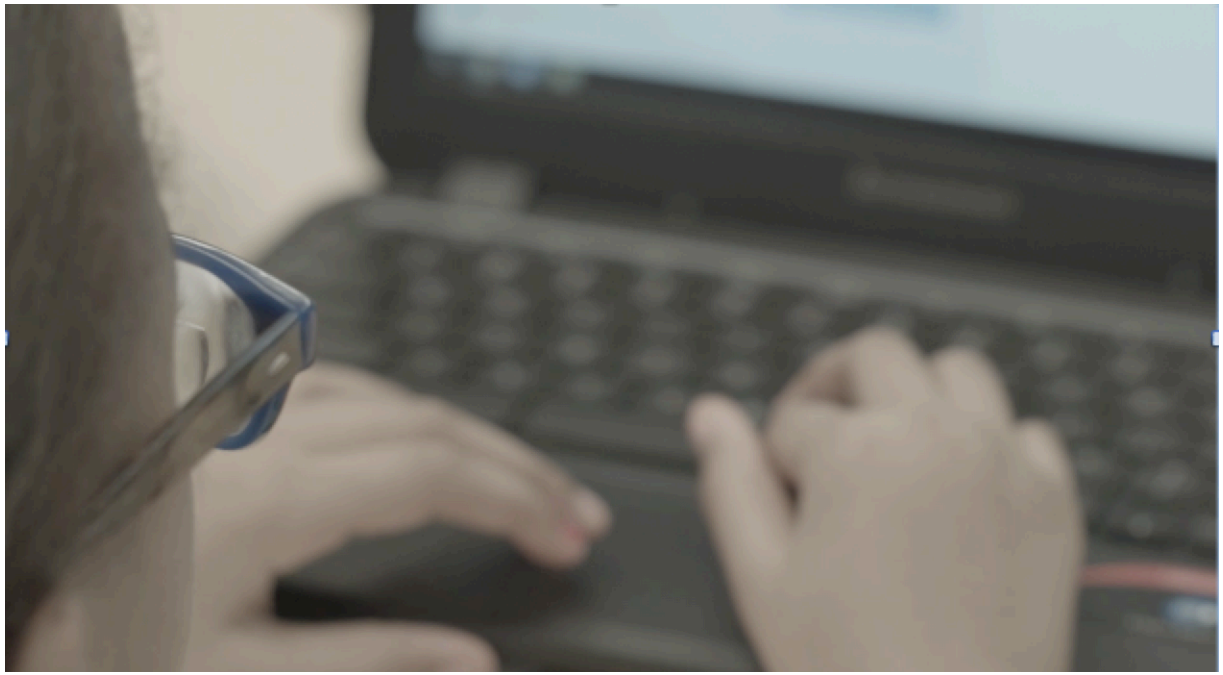
- Study instructional approaches that have shown success in other content areas within CS education
- Work in classrooms that have cultural, socioeconomic, and academic diversity
- Attempt to integrate CS into content areas
- Work closely with teachers and administrators to see whether our research questions are relevant and important to K-12 education

Programming Platforms for Our Research

- Graphically intuitive block-based software to teach programming
- Examples: Scratch & Code.org



Video Example of Integrated Computing and Mathematics



- http://stemforall2016.videohall.com/presentations/691

Integration Research questions

1. How do elementary teachers with limited computing experience integrate computing into math instruction?

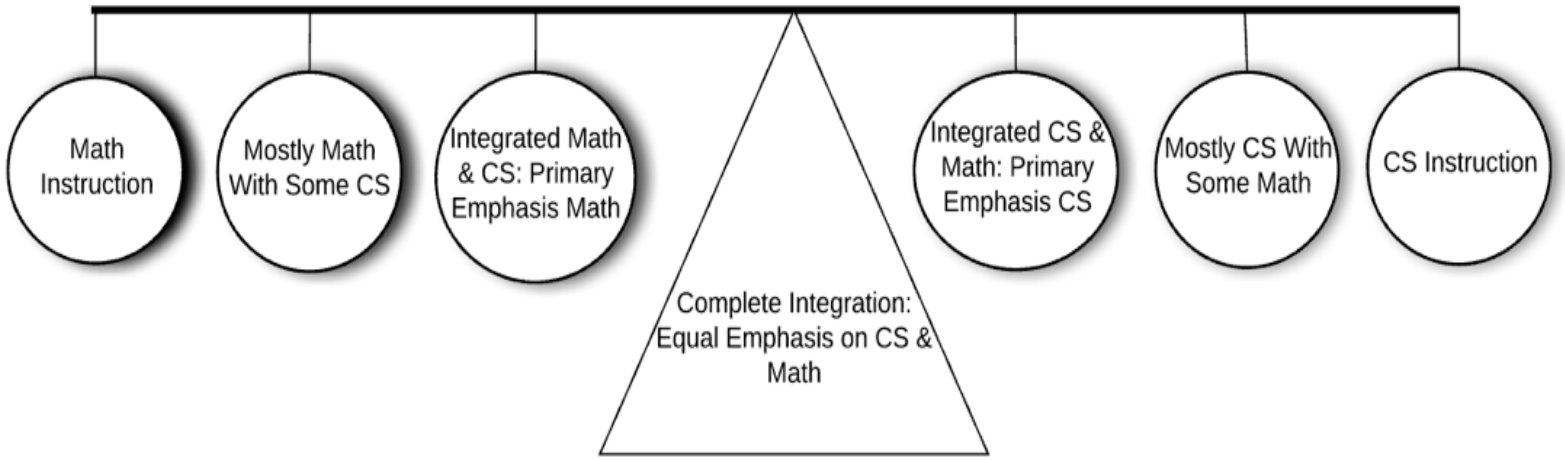
2. What challenges do elementary teachers with limited computing experiences face as they attempt to integrate computing into their math instruction?

3. How do elementary teachers with limited computing experiences support the needs of struggling learners?

Israel, Pokimica, Wherfel, & Reese (in preparation)

Integrated Instruction Conceptual framework

- Curricular integration model adapted from Kiray (2012)

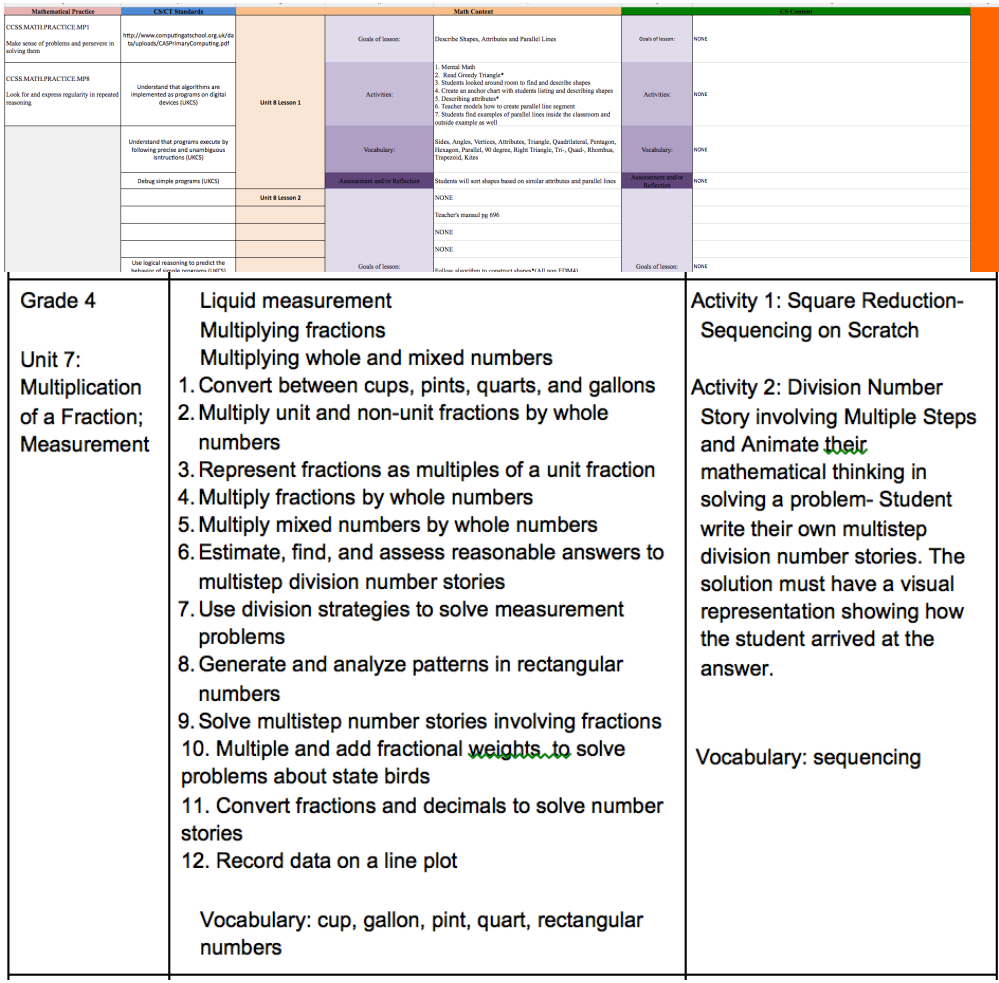


Israel, Pokimica, Wherfel, & Reese (in preparation)

Data collection methods

- Observations (of instruction & student work in Scratch)
- Lesson plans
- Teacher interviews

- Analysis: Interpretative qualitative methodology with a constant comparative analyses
  - Coding schemes & emerging themes (interviews)
  - Written summaries (observations)
  - Spreadsheet (lesson plans)
    - with themes related to math & CS/ CT content, the standards, and the balance between math and CS instructional time



Mathematical Practice	CS/CT Standards	Math Content			CS Content	
CCSS.MATH.PRACTICE.MP1 Make sense of problems and persevere in solving them	http://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf	Unit 8 Lesson 1	Goals of lesson:	Describe Shapes, Attributes and Parallel Lines	Goals of lesson:	NONE
CCSS.MATH.PRACTICE.MP8 Look for and express regularity in repeated reasoning	Understand that algorithms are implemented as programs on digital devices (UKCS)		Activities:	1. Mental Math 2. Read Greedy Triangle* 3. Students looked around room to find and describe shapes 4. Create an anchor chart with students listing and describing shapes 5. Describing attributes* 6. Teacher models how to create parallel line segment 7. Students find examples of parallel lines inside the classroom and outside example as well	Activities:	NONE
	Understand that programs execute by following precise and unambiguous instructions (UKCS)		Vocabulary:	Sides, Angles, Vertices, Attributes, Triangle, Quadrilateral, Pentagon, Hexagon, Parallel, 90 degree, Right Triangle, Tri-, Quad-, Rhombus, Trapezoid, Kites	Vocabulary:	NONE
	Debug simple programs (UKCS)		Assessment and/or Reflection	Students will sort shapes based on similar attributes and parallel lines	Assessment and/or Reflection	NONE
		Unit 8 Lesson 2		NONE		
				Teacher's manual pg 696		
				NONE		
	Use logical reasoning to predict the behavior of simple programs (UKCS)		Goals of lesson:	NONE Follow algorithm to construct shapes*(All new STEM4)	Goals of lesson:	NONE

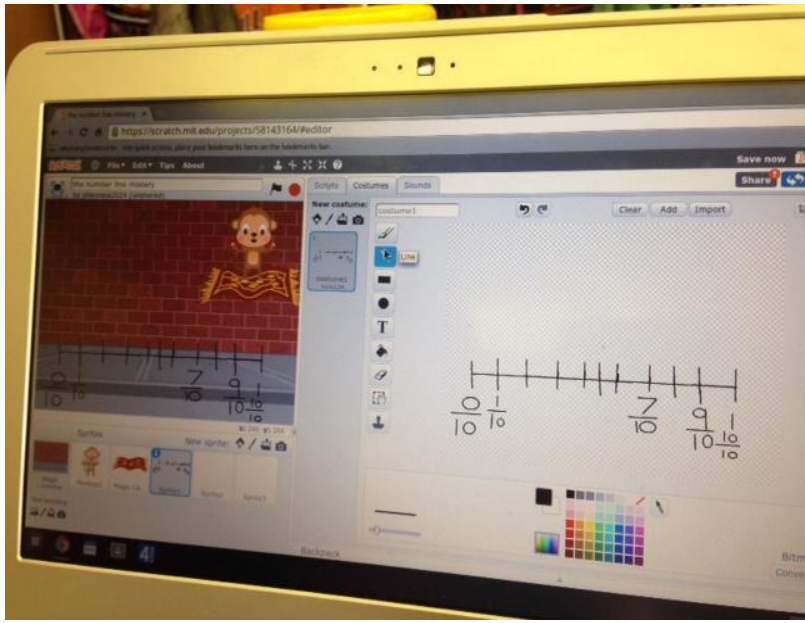
Grade 4 Unit 7: Multiplication of a Fraction; Measurement	Liquid measurement Multiplying fractions Multiplying whole and mixed numbers 1. Convert between cups, pints, quarts, and gallons 2. Multiply unit and non-unit fractions by whole numbers 3. Represent fractions as multiples of a unit fraction 4. Multiply fractions by whole numbers 5. Multiply mixed numbers by whole numbers 6. Estimate, find, and assess reasonable answers to multistep division number stories 7. Use division strategies to solve measurement problems 8. Generate and analyze patterns in rectangular numbers 9. Solve multistep number stories involving fractions 10. Multiple and add fractional weights to solve problems about state birds 11. Convert fractions and decimals to solve number stories 12. Record data on a line plot Vocabulary: cup, gallon, pint, quart, rectangular numbers	Activity 1: Square Reduction- Sequencing on Scratch Activity 2: Division Number Story involving Multiple Steps and Animate their mathematical thinking in solving a problem- Student write their own multistep division number stories. The solution must have a visual representation showing how the student arrived at the answer. Vocabulary: sequencing

Findings

- Teachers perceived the **integrated math and CS lessons as highly motivating**, especially for students who typically struggled in math

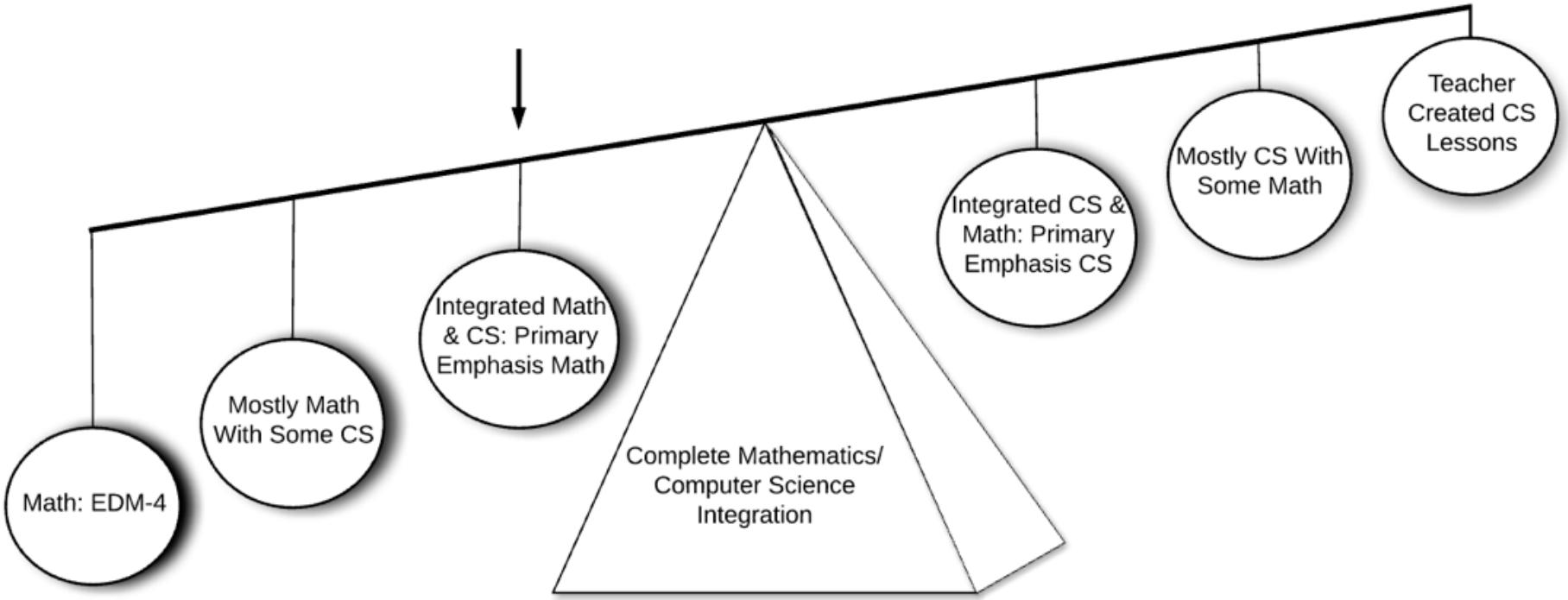
- **The math instruction was more dominant** than the CS content, and the teachers cycled between new mathematics content and new CS content so that the students only learned content in one discipline at a time – ***skewed integration model***

- **Teachers used multiple strategies to meet the needs of struggling learners** including differentiated levels of explicit instruction in new CS content and encouraged student collaboration when students had difficulty with the integrated mathematics and CS tasks.



Israel, Pokimica, Wherfel, & Reese (in preparation)

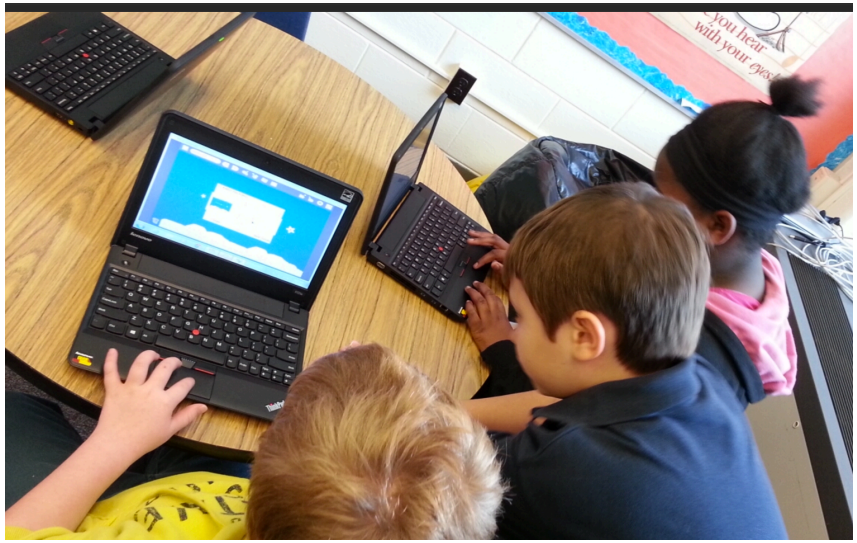
Skewed integration model



Israel, Pokimica, Wherfel, & Reese (in preparation)

Measuring Collaborative Computing

- Collaborative Computing Observation Instrument (C-COI)
  - Use video screen capture software to capture all computing activities and audio of student collaborations
  - Dependent variables include amount of time persisting on tasks, methods of help seeking, collaborative problem-solving, and computing challenges.



Israel, Shehab, & Wherfel (under review)

What can be analyzed using the C-COI?

Questions we wanted to ask:	Constructs
How does the student request help?	Adaptive vs. Negative Help Seeking
How does the student individually problem solve?	Persistence
What kind support(s) did the student receive?	Collaborative Problem-Solving
Did the computing experience result in skill/concept acquisition?	Understanding CS concepts/ vocab.

C-COI (cont.)

- Displays individual or collaborative behaviors that a student show during computing task
- Event=Sequence behaviors beginning when a student starts to work on a computing sub-task (e.g. making a sprite dance) and ending when the sub-task is solved.
- Each event can be composed of three types of paths
  - Problem Solving Path
  - Socialization path
  - Expressing curiosity, excitement, and accomplishment path

Israel, Shehab, & Wherfel (under review)

C-COI Online Version

Student-Peer Interaction

Timestamp: (mm:ss) 3 : 0 Proceed

- Peer does not know how to help
- Peer elicits another person
- Another peer joins the discussion about student problem
- Peer physically shows by taking over the students computer
- Peer verbally tells the steps explicitly
- Peer physically shows and explains at the same time
- Peer and student discuss problem
- Peer explains the problem
- Peer shares curiosity with student
- Student shares curiosity with peer
- Another peer joins the discussion about student's excitement
- Another peer joins the discussion about peer's excitement
- Other (use notes)

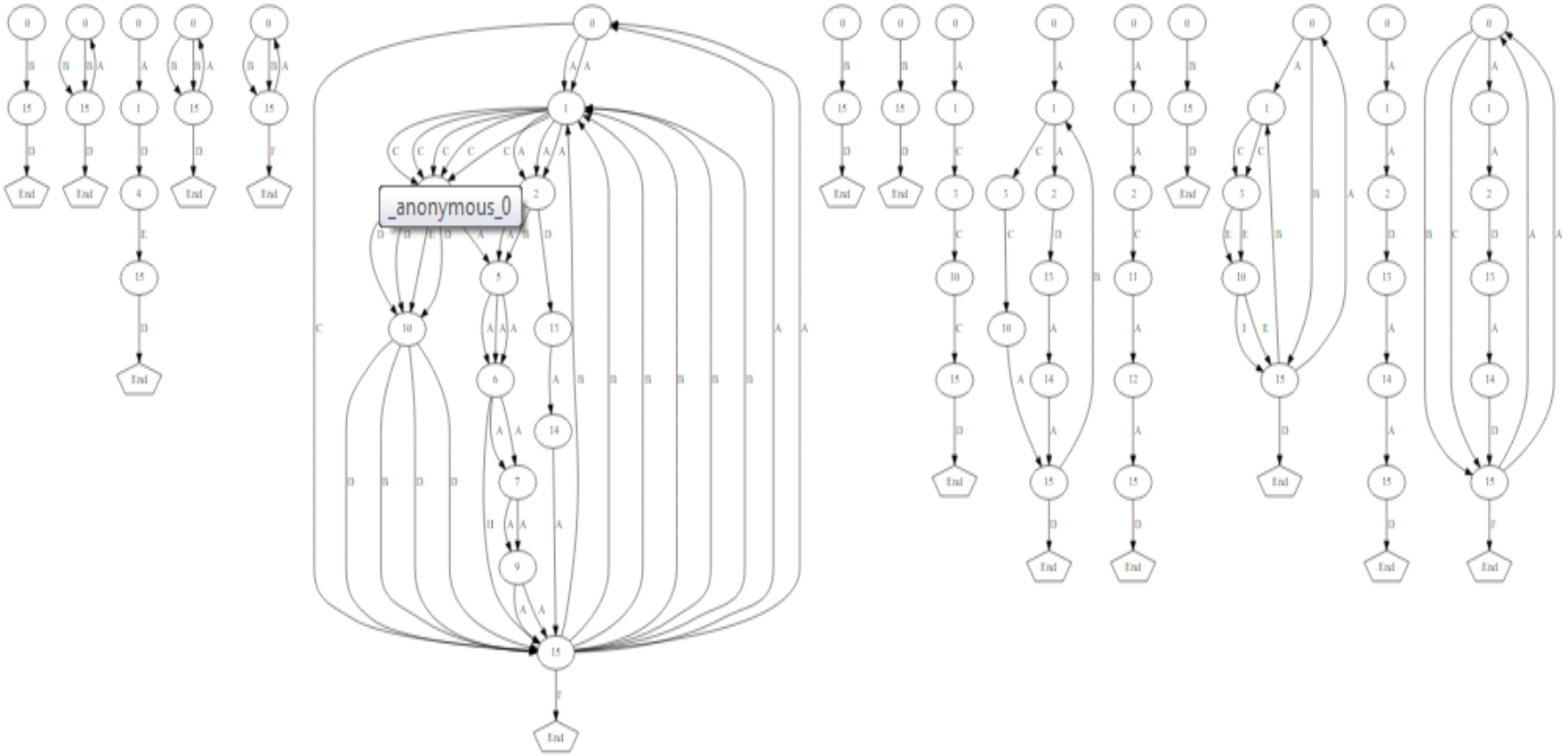
Notes:

Node 1: Capturing student's action	Node 2: Capturing student's expression	Node 3: Problem content
Student seeks attention Student is initiated by a peer Student is initiated by an adult Student offers elicited support to peer Student offers unelicited support to peer Student works independently without self-talk	Student expresses a problem Student expresses curiosity, excitement, or accomplishment Student socializes Other (use notes)	Problem is related to general technology Problem is related to computing/programming Problem is related to academic content Problem is related to navigating software Other (use notes)
Node 4: Help Seeking	**Node 5: Describing curiosity or excitement**	**Node 6: Describing socialization**
Student clearly states how they need help Student expresses need for help but is not explicit to the problem Peer offers help in response to the student's frustration Peer offers unelicited help Adult offers help in response to student's frustration Adult offers unelicited help Other (use notes)	Student is curious about something associated with their own work Student is curious about something associated with peer's work Student is excited about something associated with their own work Student is excited about something associated with peer's work Student wants to show or express accomplishment on their own work	Student socializes with peer, not related to computing Student socializes with adult, not related to computing Other (use notes)
Node 7: Peer or student's response to help seeking	**Node 8: Who is initiated?**	**Node 9: Response to initiation**
Peer helps student with a problem on student's computer Peer seeks student's curiosity/excitement/accomplishment on student's computer Peer starts socializing, heard on student's computer	Student initiates peer Student initiates adult Student dismisses their attempt to interact Other (use notes)	Peer verbally responds to the problem Peer verbally responds to student's curiosity/excitement/accomplishment Student verbally responds to peer curiosity/excitement/accomplishment Another person joins the student-peer interaction around
Node 10: Reporting Interaction	**Node 11: Adult's response**	**Node 12: Describing Interaction**
Peer does not know how to help Student and peer are interacting together on the student's problem Student terminates the interaction	Peer helps student with a problem on student's computer Peer seeks student's curiosity/excitement/accomplishment on student's computer Peer starts socializing, heard on student's computer Student helps peer with a problem on student's computer Student ignores the peer	Peer and student collaboratively discuss the problem, problem was solved Peer and student collaboratively discuss the problem, problem was not solved Peer explains the problem, problem was solved Peer explains the problem, problem was not solved
Node 13: Problem solved	**Node 14:Problem not solved**	
Problem was solved, student works independently Problem was solved, the student seeks attention from the same peer Problem was solved, the student seeks attention from a different peer Problem was solved, the student seeks attention from an adult	Problem was not solved, student works independently Problem was not solved, student seeks attention from the same peer Problem was not solved, student seeks attention from a different peer Problem was not solved, student seeks attention from an adult	

Understanding the (CCOI) Outcomes [The Graphs]

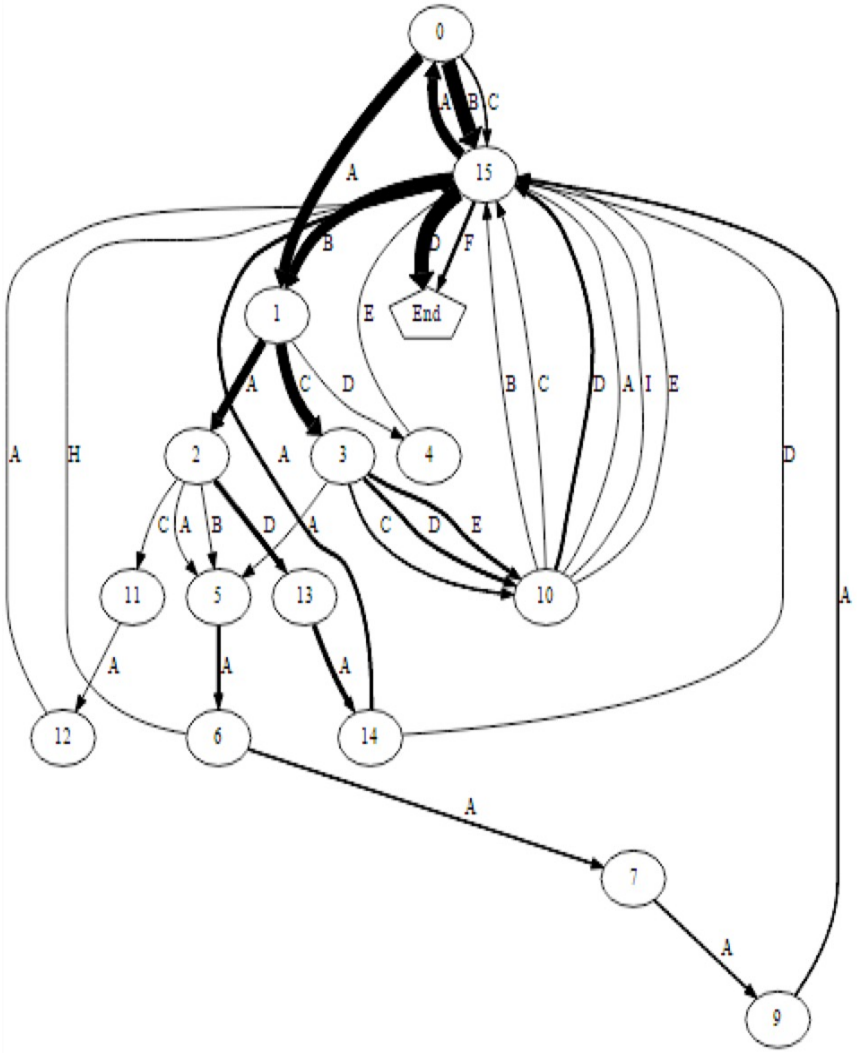
- After coding all events in a video, the different paths can be displayed as directed graphs of three different types:
  - Node Graph with Edge width (weighted)
  - Node Graph with Edge count (detailed)
  - Node Graph with separate events (detailed)

C-COI Directed Graphs
[A Node Graph, Separate]



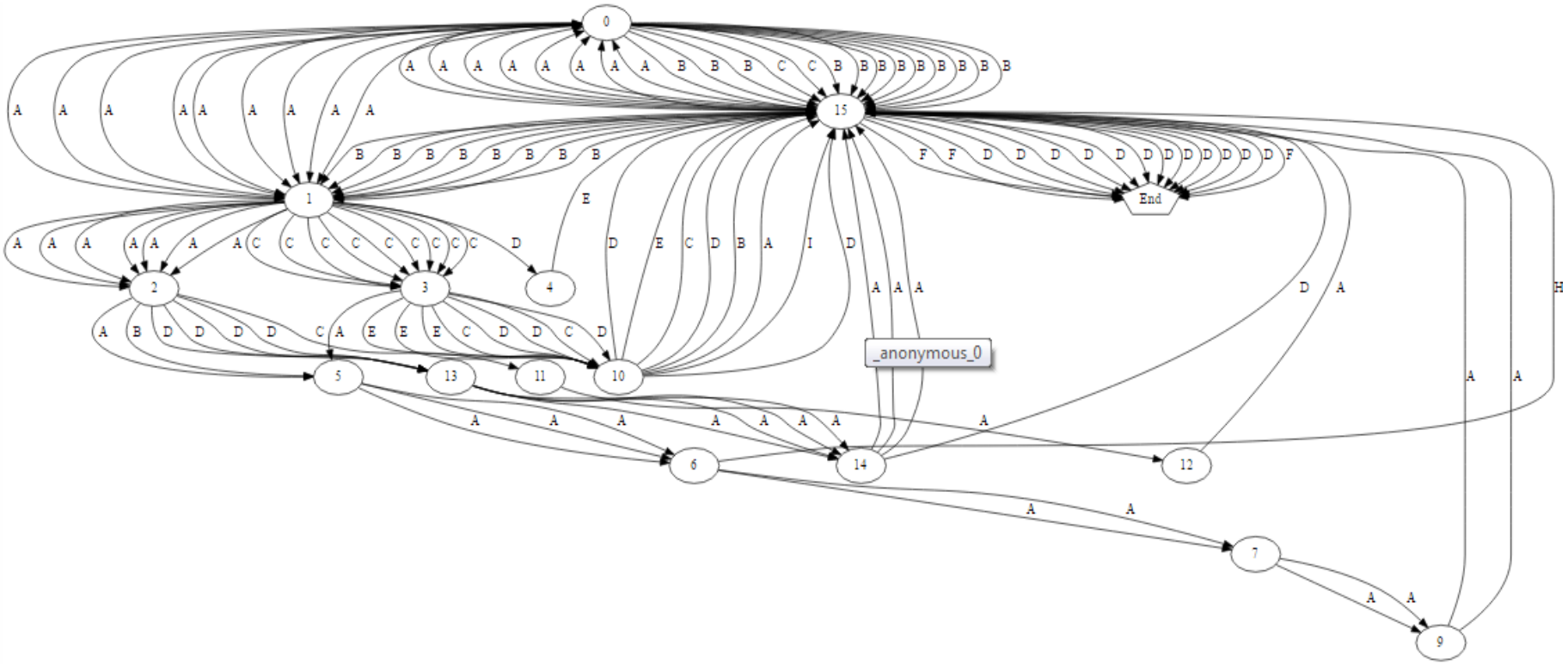
- This graph helps the researcher identify the number of events that occurred. In this case there are 15 events. Four events were solved individually (0B to 15D). There is one event that involved multiple problem solving and socialization paths but no curiosity, excitement, accomplishment paths.

C-COI Directed Graphs
[A Node Graph, Edge Width]



- What are the student's common during the three computing?
- Line 0A and 0B are both thick= student worked individually and with others during the 3 computing sessions.
- Line 1C is thicker than 1A and 1D. No Line 1B. This student was initiated by a peer (1C) more than by an adult (1D) and more than he initiated a peer (1A). The student did not initiate an adult at all (1B=0).

C-COI Directed Graphs
[A Node Graph, Edge Count]

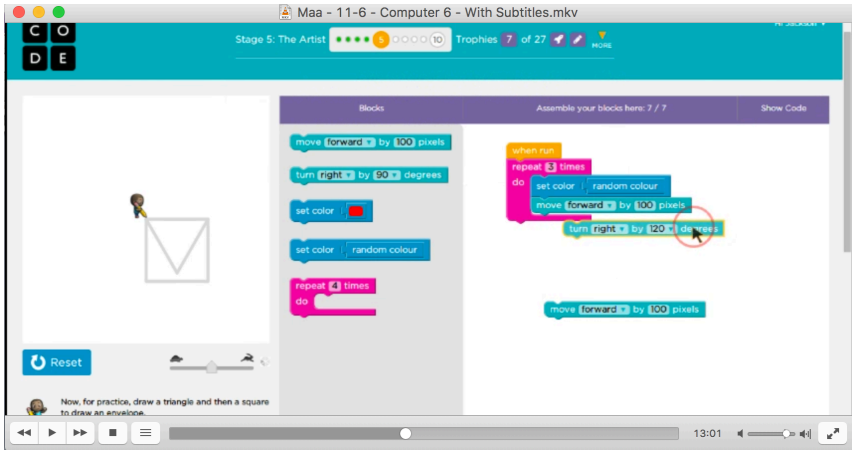


- This graph helps the researcher to count the paths.
- For example, this graph shows that the student faced three sub-tasks/ difficulties related to computing (5A=3). He interacted with a peer on two of them (6A=2). Both of them involved discussion (7A=2); however both of them were not solved (9A=2).

Video Example

- Video Example

Start at minute 13



Current Status of the C-COI

- Our team coded a diverse set of videos. Students worked in both puzzle-based (Code.org) and open-ended environments (Scratch).
- Preliminary analysis shows the following in the problem solving path:
  - Students are not explicit about their difficulties. Most times, the student says “I need help on this”.
  - Students seek help from peers & adults. Even when collaborating, they often do not succeed to solve the problem. Some persist and continue working independently until the problem is solved. Others just give up.
  - During peer discussions, often both students lack computing-specific vocabulary to explain their thoughts. Discussions usually trial and error problem solving process.

Promising Instructional Practices

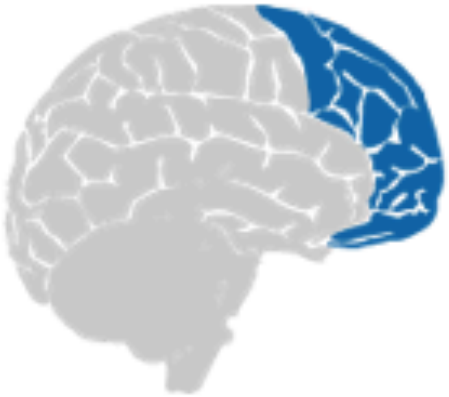
- Universal Design for Learning
- Balance explicit instruction with open inquiry
- Collaborative problem solving

Universal Design for Learning

Recognition Networks
The "what" of learning

How we gather facts and categorize what we see, hear, and read. Identifying letters, words, or an author's style are recognition tasks.

Strategic Networks
The "how" of learning



Planning and performing tasks. How we organize and express our ideas. Writing an essay or solving a math problem are strategic tasks.

Affective Networks
The "why" of learning

How learners get engaged and stay motivated. How they are challenged, excited, or interested. These are affective dimensions.

CAST.org

Universal Design for Learning (UDL)

- Provide computing instruction using multiple means of representation (e.g., pictorial representations, multimedia)
- Provide options for students to demonstrate understanding in multiple formats
- Allow students to engage with the material in different ways

CAST.org

Historic and Current Context

History of UDL

- Began in architecture with physical accessibility (e.g., curb cuts, automatic doors)
- Movement towards cognitive accessibility of instructional materials and delivery (e.g., text to speech for digital text)

Current Movement in UDL

- Every Student Succeeds Act (reauthorization of the ESEA/NCLB)
  - UDL and assessment
  - **UDL and technology adoption***
- National Education Technology Plan (2016)
  - Equity and access

ESSA language related to UDL

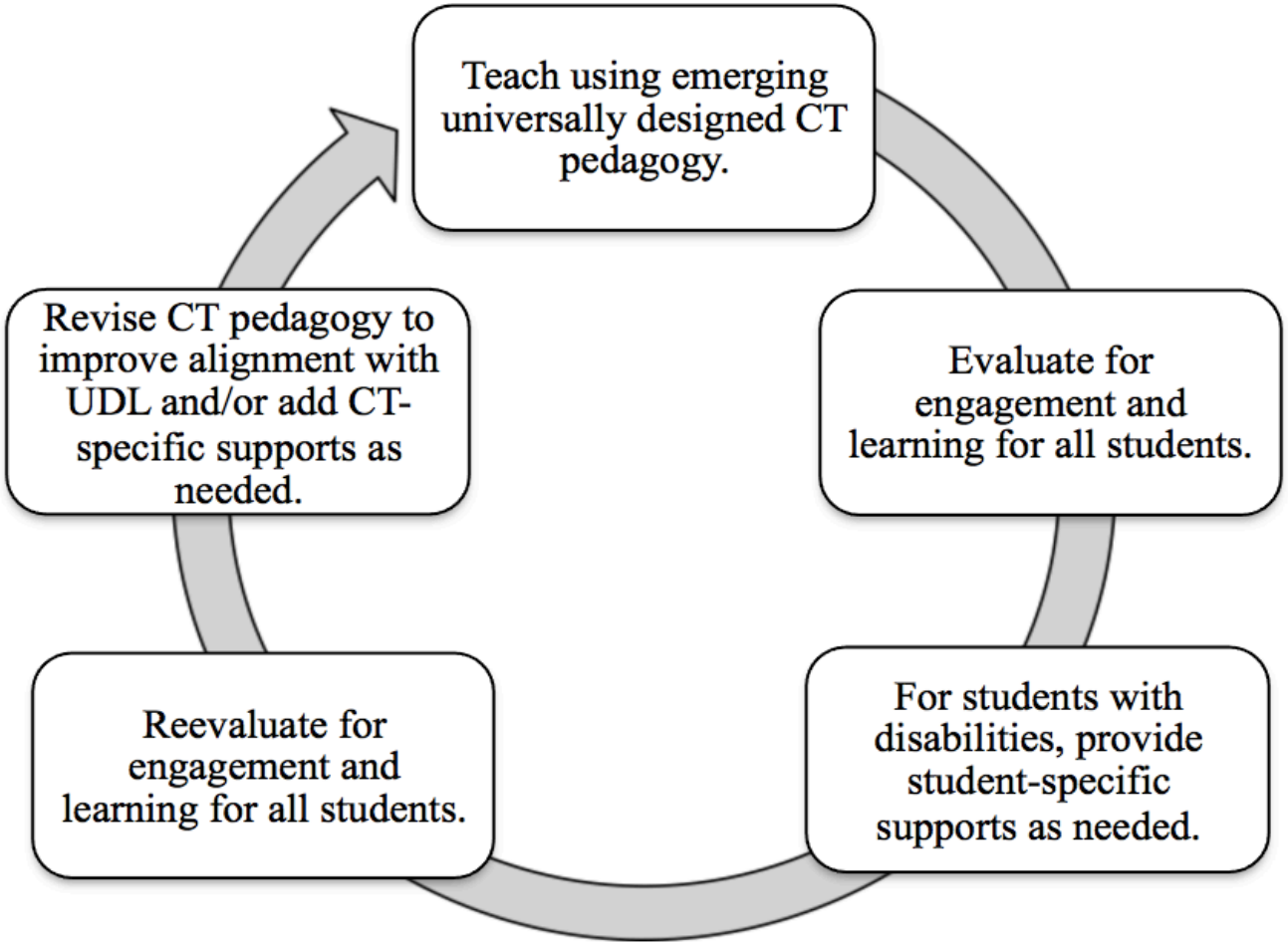
- Encourages school districts to "**use technology**, consistent with the principles of universal design for learning, to support the learning needs of all students, including children with disabilities and English learners" (P. L. 114-95, Title IV, Sec. 4104(3)(c)(i)(II)).

Examples of UDL in CS

Israel et al. (2015)

Multiple Means of **Representation**	Multiple Means of **Action and Expression**	Multiple Means of **Engagement**
1. Provide options for perception -Model computing lessons using an interactive whiteboard, videos, or already created templates -Give access to modeled code while students work independently -Provide access to video tutorials of computing tasks	*4. Provide options for physical action* -Provide teacher's codes or use partially completed code as templates -Include unplugged activities that teach through physical representations of abstract computing concepts -Use assistive technology including larger/smaller mice, touch-screen computers, screen readers	*7. Provide options for recruiting interest* -Give students choices (project, topic, display of project). -Allow students to make projects relevant to culture and age -Minimize common "pitfalls" for both computing and integrated content by considering barriers to learning from the beginning of the planning process.

UDL and CS/CT



Snodgrass, Israel, & Reese (2016)

Balancing Explicit Instruction and Open Inquiry

- Computing is inherently open-ended, complex, and student-driven.
- Explicit instruction is a systematic and direct way of teaching. This is teacher directed.
- Can we balance these two approaches?

Examples of Explicit Instruction in CS Education
(Israel et al., 2015)

Explicit Instruction	Definition in CS Education	Example
Focus instruction on critical content	Teach skills & concepts associated with CS ideas	Decide which CS skills to teach (e.g., using conditionals to create an animated story)
Provide step-by-step demonstration to break down complex tasks	Model procedures including think-alouds	Model a particular code (such as using conditionals) step-by-step with examples
Provide numerous opportunities for practice	Provide more scaffolding initially and reduce those over time	Include supports (such as guiding questions) as students try coding. Encourage risk taking and independent problem solving.

Balancing Explicit Instruction and Inquiry Example

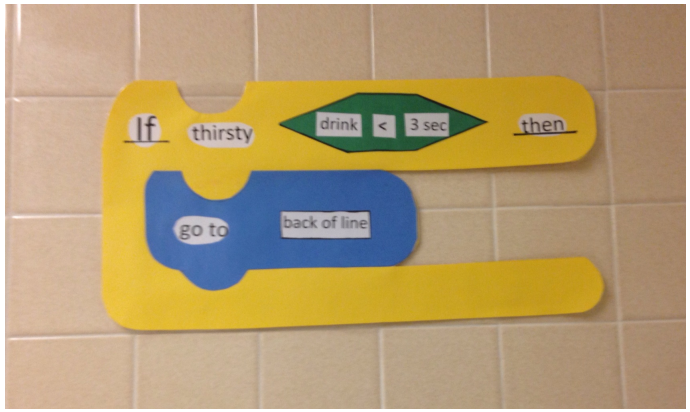
- Teachers at a local school (Kenwood) wanted to teach kids the concept of conditionals:
  - Expressions that are evaluated as true or false to determine program flow. These are logic statements.
    - If a condition is true, then do one thing
    - If a condition is false, do another thing
  - Step 1: worksheet with “If/then” statements to use in real world situations
  - Step 2: Play a card game to practice conditional statements
  - Step 3: Program in Scratch using “If/then” statements

Example (cont.)

- Write different conditionals that you do at school or home

If <_____> then

[_____]



Encouraging Student Collaboration

- Computing can be inherently collaborative.
- Our research has shown:
  - Help seeking/giving
  - Collaboration to solve problems
  - Collaboration to share excitement, pride, work
- But….we need to teach kids effective collaboration strategies

Findings and Tips

1. Some students spend a LOT of time persisting, do not collaborate, and do not successfully complete the task.

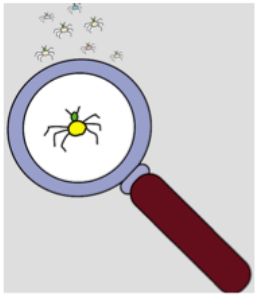
   Collaborating interrupts flow

2. Most common collaborative events ended with problems not solved
   a. Students are not effectively using the collaborative script to solve the problem
   b. Students are not watching the video hints
   c. Students lack understanding of the computer science concepts that are associated with the problem

Tools to Help Collaborative Discourse

Debugging Detective Questions:

- What happened when I ran my code?
- What did I want my code to do?
- Does any part of my code work?
- Do I know where the problem is in my code?



Debugging Detective Questions
What do I do when I am stuck?
Finding the "suspect" in our code.

Using these questions, students are encouraged to work with friends to solve the problem.

Next Steps

- Study in NYC schools using different computing platforms: Codable, Bootstrap, and Code.org for students at risk for academic failure
- NSF STEM+C project is started in January of 2016 to study integrated computing and math instruction http://everydaycomputing.org
- Individual and content-specific supports students with disabilities in CS
- Continued exploration of collaborative computing

For More Information:

misrael@illinois.edu

@misrael09

References

- Israel, M., Ramos, E., Wherfel, Q. M., & Shehab, S. (2015) Collaborative Computing Observation Instrument (C-COI). Board of Trustees of the University of Illinois at Urbana-Champaign. Available at http://mste.illinois.edu/c-coi/.

- Israel, M., Wherfel, Q., Pearson, J., Shehab, S., & Tapia, T. (2015). Empowering K-12 students with disabilities to learn computational thinking and computer programming. *TEACHING Exceptional Children, 48*(1), 45-53.

- Ladner, R., & Israel, M. (in press). "For all" in "computer science for all". *Communications of the ACM.*

- Kiray, A. S. (2012). A new model for the integration of science and mathematics: The balance model. *Energy Education Science and Technology Part B: Social and Educational Studies, 4,* 1181-1196.

- Snodgrass, M. R., Israel, M. & Reese, G. (2016). Instructional supports for students with disabilities in K-5 computing: Findings from a cross-case analysis. *Computers & Education.* Early online release. doi:10.1016/j.compedu.2016.04.011