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
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/](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

1,000,000 Unfilled Jobs by 2020

Computer Occupations 62%

All other math and science jobs 40%

Computing Jobs 60%

$500 billion opportunity

1,000,000 unfilled computing jobs

400,000 computer science graduates

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).
“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.


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
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

Math: EDM-4

Mostly Math With Some CS

Integrated Math & CS: Primary Emphasis Math

Integrated CS & Math: Primary Emphasis CS

Mostly CS With Some Math

Teacher Created CS Lessons

Complete Mathematics/Computer Science Integration

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?

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

Timestamp: (mm:ss) 3:0

- 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:
<table>
<thead>
<tr>
<th>Node 1: Capturing student’s action</th>
<th>Node 2: Capturing student’s expression</th>
<th>Node 3: Problem content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student seeks attention</td>
<td>Student expresses a problem</td>
<td>Problem is related to general technology</td>
</tr>
<tr>
<td>Student is initiated by a peer</td>
<td>Student expresses curiosity, excitement, or accomplishment</td>
<td>Problem is related to computing/programming</td>
</tr>
<tr>
<td>Student is initiated by an adult</td>
<td>Student socializes</td>
<td>Problem is related to academic content</td>
</tr>
<tr>
<td>Student offers elicited support to peer</td>
<td>Other (use notes)</td>
<td>Problem is related to navigating software</td>
</tr>
<tr>
<td>Student offers unelicited support to peer</td>
<td>Other (use notes)</td>
<td>Other (use notes)</td>
</tr>
<tr>
<td>Student works independently without self-talk</td>
<td>Other (use notes)</td>
<td>Other (use notes)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 4: Help Seeking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student clearly states how they need help</td>
</tr>
<tr>
<td>Student expresses need for help but is not explicit to the problem</td>
</tr>
<tr>
<td>Peer offers help in response to the student's frustration</td>
</tr>
<tr>
<td>Peer offers unelicited help</td>
</tr>
<tr>
<td>Adult offers help in response to the student's frustration</td>
</tr>
<tr>
<td>Adult offers unelicited help</td>
</tr>
<tr>
<td>Other (use notes)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 5: Describing curiosity or excitement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student is curious about something associated with their own work</td>
</tr>
<tr>
<td>Student is curious about something associated with peer's work</td>
</tr>
<tr>
<td>Student is excited about something associated with their own work</td>
</tr>
<tr>
<td>Student is excited about something associated with peer's work</td>
</tr>
<tr>
<td>Student wants to show or express accomplishment on their own work</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 6: Describing socialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student socializes with peer, not related to computing</td>
</tr>
<tr>
<td>Student socializes with adult, not related to computing</td>
</tr>
<tr>
<td>Other (use notes)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 7: Peer or student’s response to help seeking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer helps student with a problem on student's computer</td>
</tr>
<tr>
<td>Peer seeks student’s curiosity/excitement/accomplishment on student's computer</td>
</tr>
<tr>
<td>Peer starts socializing, heard on student's computer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 8: Who is initiated?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student initiates peer</td>
</tr>
<tr>
<td>Student initiates adult</td>
</tr>
<tr>
<td>Student dismisses their attempt to interact</td>
</tr>
<tr>
<td>Other (use notes)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 9: Response to initiation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer verbally responds to the problem</td>
</tr>
<tr>
<td>Student verbally responds to student's curiosity/excitement/accomplishment</td>
</tr>
<tr>
<td>Another person joins the student-peer interaction around</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 10: Reporting Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer does not know how to help</td>
</tr>
<tr>
<td>Student and peer are interacting together on the student's problem</td>
</tr>
<tr>
<td>Student terminates the interaction</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 11: Adult’s response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer helps student with a problem on student's computer</td>
</tr>
<tr>
<td>Peer seeks student’s curiosity/excitement/accomplishment on student's computer</td>
</tr>
<tr>
<td>Peer starts socializing, heard on student's computer</td>
</tr>
<tr>
<td>Student helps peer with a problem on student's computer</td>
</tr>
<tr>
<td>Student ignores the peer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 12: Describing Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer and student collaboratively discuss the problem, problem was solved</td>
</tr>
<tr>
<td>Peer and student collaboratively discuss the problem, problem was not solved</td>
</tr>
<tr>
<td>Peer explains the problem, problem was solved</td>
</tr>
<tr>
<td>Peer explains the problem, problem was not solved</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 13: Problem solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem was solved, student works independently</td>
</tr>
<tr>
<td>Problem was solved, the student seeks attention from the same peer</td>
</tr>
<tr>
<td>Problem was solved, the student seeks attention from a different peer</td>
</tr>
<tr>
<td>Problem was solved, the student seeks attention from an adult</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node 14: Problem not solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem was not solved, student works independently</td>
</tr>
<tr>
<td>Problem was not solved, student seeks attention from the same peer</td>
</tr>
<tr>
<td>Problem was not solved, student seeks attention from a different peer</td>
</tr>
<tr>
<td>Problem was not solved, student seeks attention from an adult</td>
</tr>
</tbody>
</table>
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.
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).
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)

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

Teach using emerging universally designed CT pedagogy.

Revise CT pedagogy to improve alignment with UDL and/or add CT-specific supports as needed.

Evaluate for engagement and learning for all students.

Reevaluate for engagement and learning for all students.

For students with disabilities, provide student-specific supports as needed.
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)

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

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](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


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