

Machine Analysis of Interaction Histories in Classroom Learning Partner

6.UAP Final Report

Adisa Kruayatidee

Advisor: Dr. Kimberle Koile, MIT Center for Educational Computing Initiatives

22 May 2015

1 Introduction

Although technology has become an integral part of our everyday lives, classrooms have found few new ways to incorporate these advances. This disconnect between technology and education continues to be a growing problem, as students miss out on learning needed technological skills, and teachers struggle to manage increasing class sizes. The ultimate goal for those concerned with this disconnect is to usher technology into the educational setting in a way that incites curiosity in children to learn and facilitates better student-teacher communication.

Classroom Learning Partner (CLP) [1-5, 7] is a classroom interaction system at the forefront of that mission. CLP replaces traditional paper workbooks with tablet-based digital notebooks, so students can wirelessly submit work and receive teacher feedback in real time. Given this real-time access to student work, one of the challenges is enabling the teacher to efficiently grasp the students' understanding of the material. Teachers can get insight into what their students understand by viewing the student work as it is submitted. Often, however, the process of creating the work reveals valuable information that is not visible in a final artifact. Did a student work straightforwardly and quickly? Did a student seem to struggle with several false starts before creating an artifact or arriving at an answer? Being able to replay what we can an *interaction history* to show the steps a student took when

working on a problem can help a teacher answer questions such as these, but is quite time-consuming, especially when students may submit multiple versions of their work, e.g., when they realize they have made a mistake.

This 6.UAP project provides one solution to this challenge: a program that will provide a summary of a student's process of creating an artifact. The artifact for the current research is a representation used when learning multiplication and division in upper elementary grades. The summary maps low-level actions a student carries out with a tablet pen, e.g., making an ink stroke, into higher level actions that are relevant from a math education point of view, e.g., creating a group of ink strokes along the right edge of an array – an action that might be identified as skip counting, e.g., 8, 16, 24, 32. The program takes as input a list of low-level actions, which we call *history items*, and produces a list of higher level actions, which we call *coded actions*. The highest level of coded actions were defined by a team of researchers through a process of hand-coding a corpus of student work. The program's goal is to produce as many of the coded actions automatically as possible.

2 Background

CLP operates in a classroom on a set of wirelessly connected table computers. It can be used with any age student; the current research focuses on upper elementary classrooms. It employs a notebook metaphor, and students use CLP as they would a paper notebook, by writing and drawing on “pages”. Students also use tools provided by CLP to create visual representations. For upper elementary students learning multiplication and division these tools include array and number line, which are analogous to their paper versions, as well as new tools that we call stamp and division template. (For a discussion of the CLP tools see [3-7].) Students use the tablet pen to draw and interact with the tools, then send their pages wirelessly to the teacher. The teacher then can view and sort

through the pages, identifying students who may need help and selecting examples of work as the focus of class discussion.

2.1 CLP Interactions

Students interact with a personal CLP notebook via a tablet pen or their finger. By strictly having this type of interaction with no keyboard usage, CLP aims to preserve and take advantage of student familiarity with paper notebooks and drawing, while simultaneously enhancing the student's interaction experience. The pen is especially important for the kinds of STEM (Science, Technology, Engineering, Math) activities that CLP research targets, since many of these activities involve a combination of text and drawing. Students, for example, may work on a math problem and draw a picture to explain the answer, or collect and record data on the tablet, then graph the data. Figure 1 shows a student using the pen to draw jumps on a number line when solving a multiplication problem.

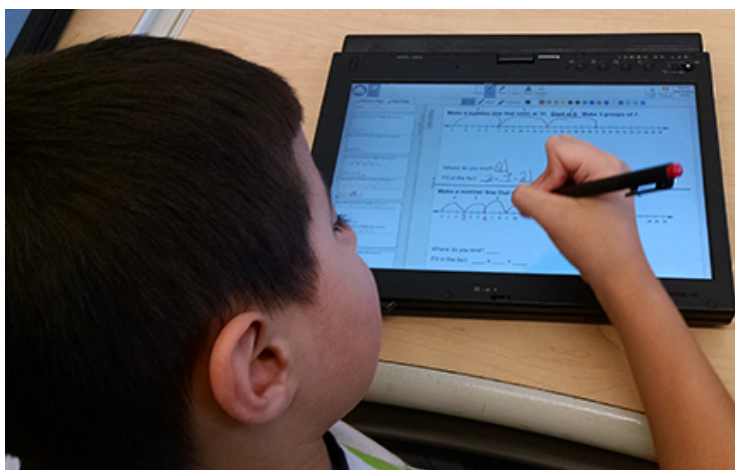


Figure 1: A third grade student using his CLP Notebook

In addition to offering a richer student-teacher interaction experience, CLP aims to be a tool that allows teachers to more easily and quickly comprehend how well students understand the material being taught. CLP stores full replays of the student activity on each page, which the teacher can review later. The teacher, however, may not have time to replay every submission. Functionality that will automatically analyze submitted student work and summarize results will give the teacher insight into student understanding and leave the teacher with more time to focus on the students themselves.

2.2 Arrays

Arrays are one of the interactive elements students can create using a CLP tool. An array is a rectangular grid of unit boxes that can be used to represent a product as an area or a number of groups of a particular size. A row, for example, may represent a group; the number of columns then represents the number of objects in that group. To create an array in CLP, a student specifies dimensions, and a rectangle of the correct proportions appears on her page. Interactions with the array, e.g., cutting an array or snapping arrays together, enable students to create structured representations that can be interpreted, both by human and by machine. Arrays are one of two CLP elements analyzed in the examples in this paper.

When arrays are used to teach multiplication, information about what an array represents is useful for automatic interpretation, as the rows and columns of the array can be manipulated in various ways. In Figure 2, the student is using arrays to visualize multiples of objects and to keep track of smaller arithmetic equations as she breaks down the big problem.

Extended Response

Solve. Show how you find the answer.

12. A spider has 8 legs and a butterfly has 6 legs.
How many legs do 4 spiders and 9 butterflies have in all?

They have _____ legs in all.

Figure 2: Array usage example

A common way students use arrays in multiplication is by skip counting down the side, as shown in Figure 2. Skip counting is a way to find the number of units in an array by counting by group size. It is useful when students understand the concept of multiplication and want to use a representation to help them find a product that they may not already know as a multiplication fact.

2.3 Number Lines

Number lines are another interactive element provided by CLP. A number line is a line with unit ticks, labeled at intervals. Typically, students will make jumps on a number line to count groups of objects of a specified size, with the length of each jump representing the size of a group. In CLP, students specify the length of a number line, and CLP automatically labels the ticks. Students use the tablet pen to draw jumps on the number line as in Figure 3. CLP then labels each jump with the size of the jump.

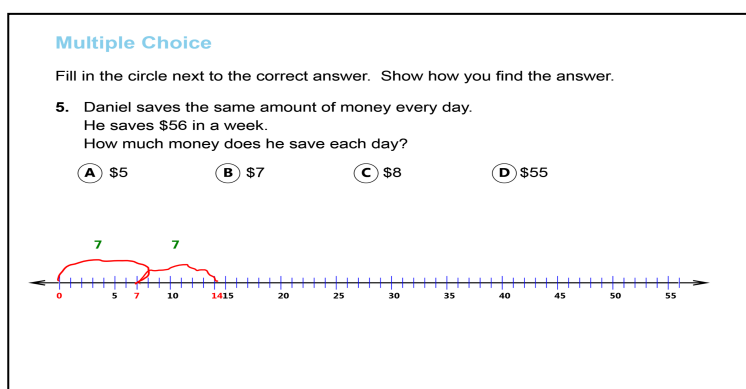


Figure 3: Number Line usage example

Number lines can be used in either multiplication or division problems. In either case, they are used to represent a certain number of groups of a given size: A single jump can represent a group; the length of the jump then represents the group size; the ending point of the last jump represents the product or dividend. In multiplication problems, the product is unknown, and students estimate the length of a number line then draw a specified number of jumps of a given size to find the product. In division problems, the dividend is used to specify the length of a number line, and the divisor is used as jump size. The student then draws jumps of the specified size until reaching the end of the number line. The number of jumps represents the quotient. A remainder is represented by the tick marks between the last jump and the end of the number line.

With CLP's tools, a student creates visual representations, then uses the pen to annotate and interact with the representations, as with array skip counting or number line jumping.

2.4 Page Replay

CLP stores both the final representation and the student's interaction history, which provides information about the student's process of creating the representation. Shown in Figure 4 are screenshots from the replay of a page of one student's work. The goal of CLP's history interaction analysis program is to produce a list of coded actions that is as close as possible to the human-produced list. This list for the example shown in Figure 3 is: ANS MC [B: 7, I], NL [56], NL jump [56: 7, 0-56], ANS MC erase [B:7, I], ANS MC [C: 8, C]. The codes indicate that the student marked multiple choice answer B, which had a value of 7 and was an incorrect answer, created a number line of length 56, jumped by 7 from 0 to 56 on the number line, erased his multiple choice answer, then marked the correct multiple choice answer of C, which had a value of 8. The next section of this paper will continue using this page as an example to describe the development of CLP's analysis program.

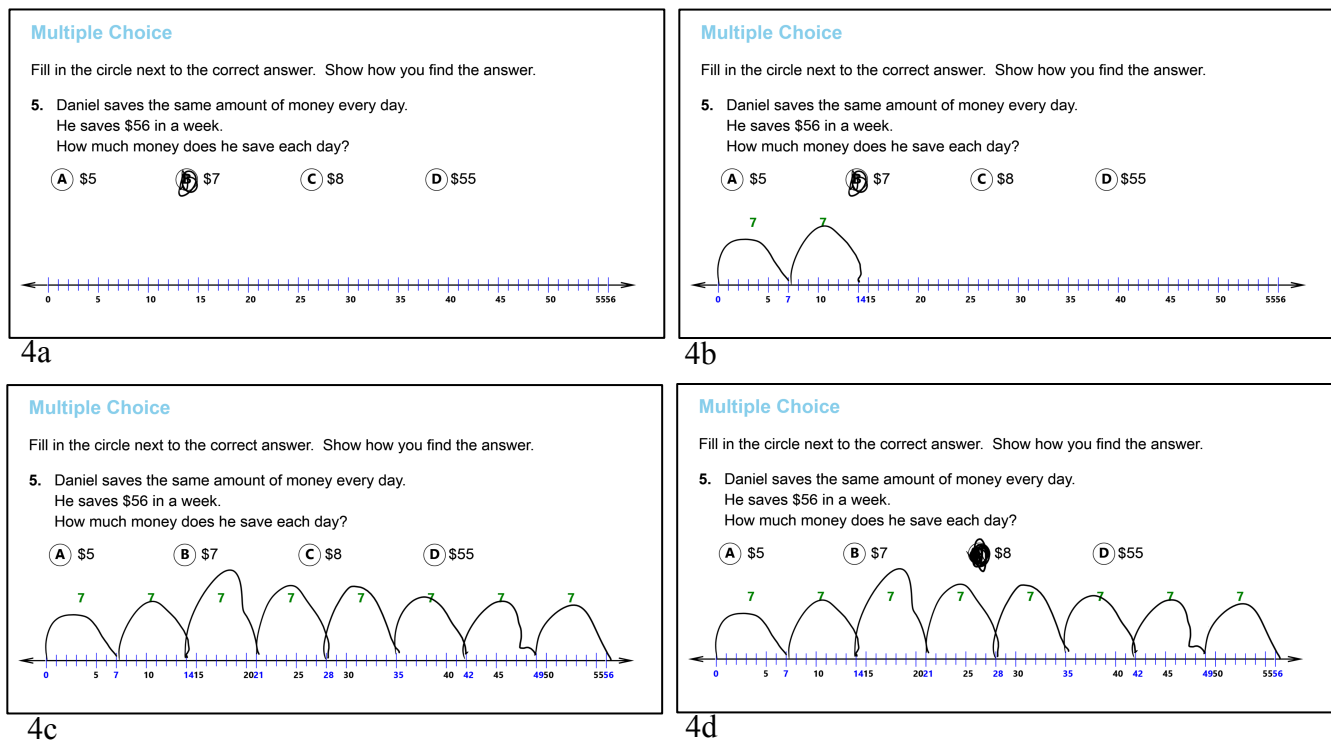


Figure 4: Jay p. 6 screenshots a-d show the process

3 Technical Approach

CLP stores information about student's interaction on a page: Each time a student makes an ink stroke, creates an object, or manipulates an object, CLP records each interaction as what we call a *history item*, which can be considered the smallest unit of change on a page. When work on this project began, there was not a way to summarize an entire page's interaction, which is represented as a list of history items, and viewing a very long list of individual interactions, e.g., a single ink stroke for the vertical component of a "4", would not provide much insight to a teacher. The analysis program developed for this UAP project starts off with the list of history items, which is used to generate the replays, and manipulates and makes several passes through the data in order to produce a list of human-understandable coded actions.

CLP is written in C# and uses the Windows Presentation Foundation for its development platform. The system has been used on Lenovo ThinkPad X Series tablets with both touch and pen input enabled and on Microsoft Surface Pro tablets.

3.1 Initial Development

To better understand the history items and to gauge the next steps of the project, the first step was to write a routine to translate the list of history items into a human-readable format. For the page example shown above in Figure 4, this routine produced the following list:

Index # 0, Added 1 stroke(s).

Index # 1, Added NumberLine [56].

Index # 2, Moved NumberLine [56].

Index # 3, Added 1 jump of 7 from 0 to 7 on number line [56].

Index # 4, Added 1 jump of 7 from 7 to 14 on number line [56].

Index # 5, Added 1 jump of 7 from 14 to 21 on number line [56].

Index # 6, Added 1 jump of 7 from 21 to 28 on number line [56].

Index # 7, Added 1 jump of 7 from 28 to 35 on number line [56].

Index # 8, Added 1 jump of 7 from 35 to 42 on number line [56].

Index # 9, Added 1 jump of 7 from 42 to 49 on number line [56].

Index # 10, Added 1 jump of 7 from 49 to 56 on number line [56].

Index # 11, Added 1 strokes.

Index # 12, Removed 1 stroke(s).

Index # 13, Removed 1 stroke(s).

Index # 14, Added 1 strokes.

Index # 15, Added 1 strokes.

Index # 16, Added 1 strokes.

This format is readable, but not quickly. For example, having an individual line for each number line jump is cumbersome; it would be better to group them into a single statement. Similarly, we could group consecutive ink actions of the same type, as they are often part of the same group of work, e.g. writing an equation or filling in an answer.

In addition, the codes used by the researchers involve locations of ink groups on the page. This location information is important because ink over an object, versus to the side of it, implies different things about the semantics of the ink. Ink on the side is likely computation, for example, whereas ink over the object is likely label information, e.g. how many objects it represents. Keeping track of ink locations will be useful for future work involving handwriting and ink analysis.

3.2 HistoryAction Classes

After analyzing the human codes and comparing them with CLP's existing history item classes, the next step was to create new classes called *history action classes*, for ease of translating into action codes. Based on how they are represented in the human codes, history items were sorted into different history action classes. Each history action instance stores formatted information needed in the coded action, such as the dimensions of the array involved in the action, so it can easily be used in the code without needing to re-implement the logic for each object and action.

The InkAction history action class was important because it allows individual history ink stroke items to be grouped. A single ink stroke does not provide much useful information, so it was necessary to recognize a pattern in multiple ink strokes, e.g. they are strokes to the right of an array.

3.3 First Pass: Converting to Coded Actions

After creating the history action classes, the next step was to write routines that would iterate through the history items list in two passes and produce the coded actions. The goal of the first pass is to translate each history item into a history action. In this pass, the ink strokes are grouped if they are not related to a CLP object, in this example the number line. If the ink strokes are associated with the number line, their extent is noted and the strokes are recorded in a single code. (The ink strokes for jumps were “captured” by the number line object based on proximity at the time that they were created.) Following is the result after the first pass:

INK change
NL [56]
NL move [56]
NL jump[56: 7, 0-56]
INK change

3.4 Second Pass: Refining Ink Actions and Locations

The second pass proved to be the most challenging phase of the development process. Each group of ink actions was to be separated by type, namely add or erase, and by location on the page relative to other objects. CLP stores a list of all objects that are used on the page, but it only stores their final positions and does not specify when each object was added or removed. Given an ink stroke, to correctly get its relative position the routine needed to figure out which objects were on the page when the stroke was added and also deduce their current positions at that time in the history.

The nature of the ink strokes in the example shown above did not result in much change

between the first and second passes, just separating added strokes from erased strokes. Final coded actions for this page are listed here, with more example pages in the next section:

INK strokes [A: over page]

NL [56]

NL move [56]

NL jump [56: 7, 0-56]

INK strokes [B: top of NL [56]]

INK strokes erase [B: top of NL [56]]

INK strokes [C: top of NL [56]]

An interesting issue revealed by this history is that the student did not originally mark the correct answer, as signified by the erase action. He filled in an answer before he made the number line, but after using the representation, corrected his answer. One difference to note is that the human codes shown earlier and repeated here, include a code for a multiple choice answer: ANS MC [B: 7, I], NL [56], NL jump [56: 7, 0-56], ANS MC erase [B:7, I], ANS MC [C: 8, C]. At the time this student work was created, CLP had no notion of a multiple choice page object, which would have enabled distinction between an ink stroke on the page and an ink stroke that designated a particular multiple choice answer. Humans can see that the ink stroke is over a multiple choice answer, but CLP doesn't "see" that distinction so its codes for the multiple choice answer are limited to particular groups of INK strokes (A, B, and C), some of which were mistakenly associated with the number line object. The multiple choice object is discussed further in the Future Work section.

4 Other Examples

In addition to the number line example, two other examples were used during the development phase.

4.1 Array Example

The first array example is shown in Figure 5. In this example the student has put down an 8x8 array, cut it into two 4x8 arrays, written equations inside the arrays, then snapped the arrays back together to

show the original array and the two sub-arrays. The human codes for this example are: ARR [8x8], ARR cut [8x8: 4x8, 4x8 a], ARR leqn [4x8 a: 4X8=], ARR arith [4x8 a: 16+16=32], ARR eqn [4x8 a: 4X8=32], ARR eqn [4x8: 4X8=32], ARR snap [4x8 a, 4x8: 8x8], ARR arith [8x8: 32+32=64], ANS MC [C 64: C]. CLP's codes for this example are shown below Figure 5.

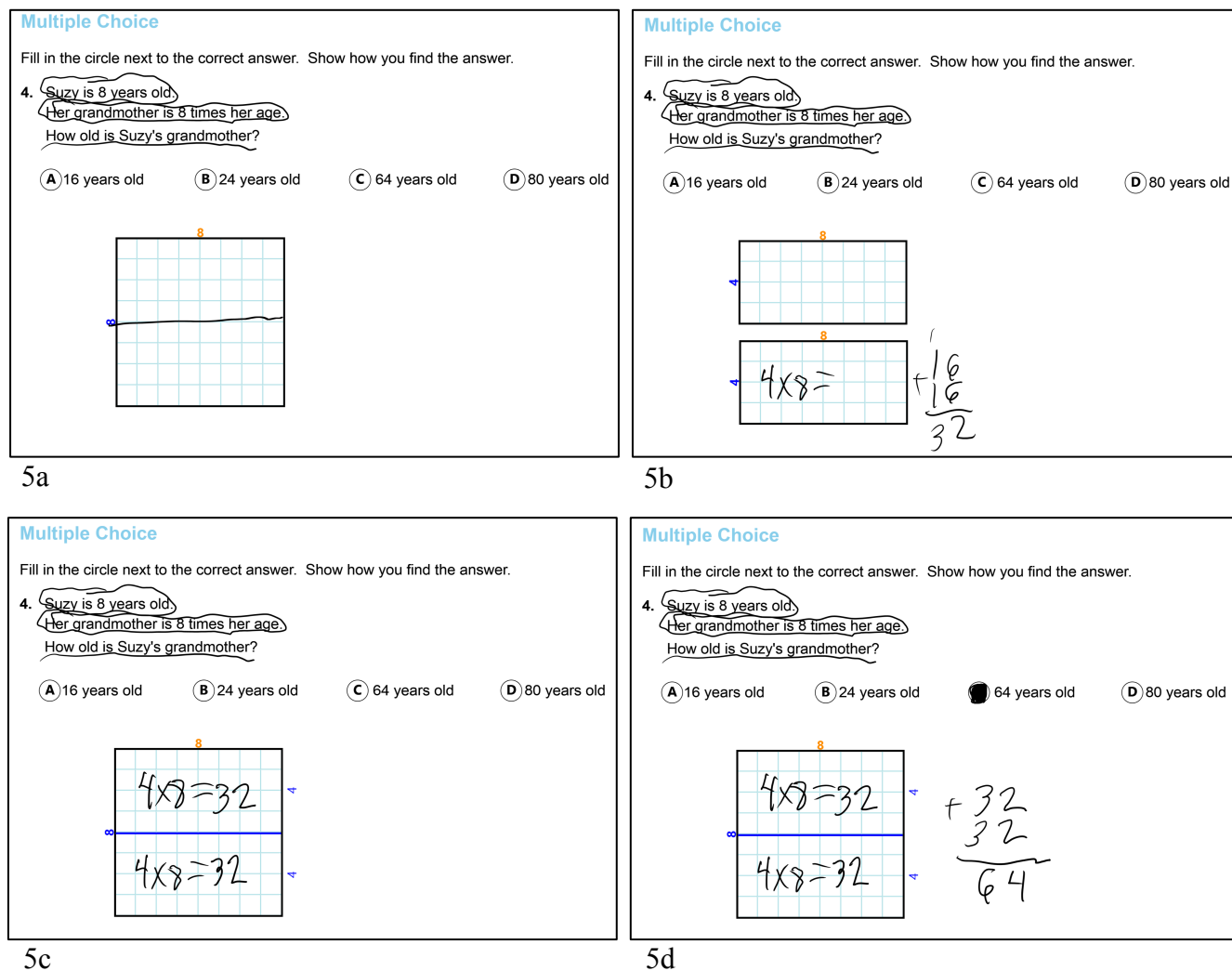


Figure 5: Justin p.5 screenshots a-d show the process

INK strokes [A: over page]

ARR [8x8]

ARR move [8x8]

ARR cut [8x8: 4x8, 4x8a]

ARR move [4x8a]

INK strokes [B: over ARR [4x8a]]

INK strokes [C: right of ARR [4x8a]]

INK strokes erase [C: right of ARR [4x8a]]

INK strokes [D: over ARR [4x8a]]

INK strokes [E: over ARR [4x8]]

INK strokes erase [E: over ARR [4x8]]

INK strokes [F: over ARR [4x8]]

ARR move [4x8a]

ARR move [4x8a]

ARR snap [4x8a, 4x8: 8x8]

INK strokes [G: top of ARR [8x8]]

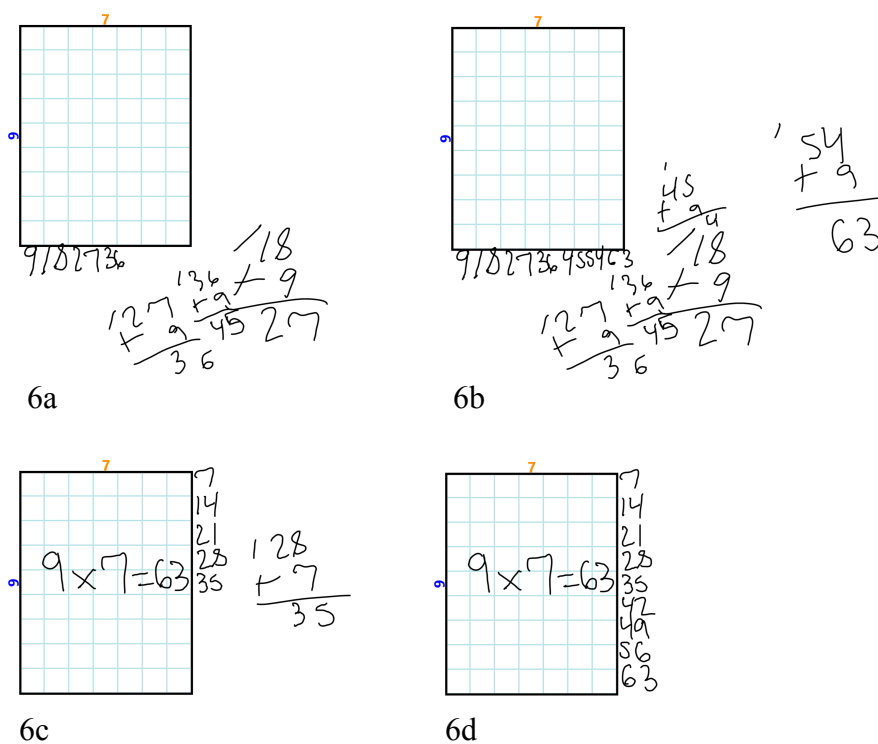
The challenge for CLP was to be able to identify the two smaller arrays uniquely and associate the ink strokes with the correct array. For example, a student may be computing an equation that is clearly related to an array, as “ $16 + 16 = 32$ ” shown next to the lower 4x8 array in Figure 5b. This relationship is obvious to a human, but one of the ink strokes (the carried 1) is physically closer to the top 4x8 array. Thus, the routine couldn't necessarily just associate each stroke with the closest object, but had to consider what made sense in the context of other strokes on the page.

Note that the human codes specify information about the ink, e.g., “4 x 8=” is the left side of an equation, designated as “leqn”. Analysis at this level of detail requires handwriting recognition routines, which were outside the scope of this project and are discussed in the Future Work section.

This example illustrates that a student's history of interaction can reveal important information not available via viewing a final artifact alone: The actions of splitting the array and using arithmetic to find answers to smaller problems indicates that even though the student did not already know the relevant multiplication facts, he was able to employ a strategy to solve the multiplication problem.

4.2 Array and Skip Counting Example

Although handwriting recognition was not within the scope of this project, interesting patterns still can be deduced from the history codes produced by the program. For example, if a student alternates between inking along an edge of an array and further away from the array, it is likely that she is performing skip counting with arithmetic. She is using the array to visualize the multiplication problem, indicating that she understands the concept, but does not yet know the product of the given factors. Shown in Figure 6 is one such example: To determine the product of 9×7 , the student skip counted along the bottom of the array, alternating with arithmetic below and to the right of the array (Figure 6a, b). Interestingly, after using skip counting to determine the product, she erased the skip counting and arithmetic, wrote the equation representing the problem, then again skip counted and used arithmetic to check her answer (Figure 6c). These steps are not evident in her final artifact (Figure 6d). Again, viewing the history of interactions reveals additional information about a student's process.



5 Results

Overall, the results from this initial implementation have been promising. The coded actions the program produces should make it much simpler and less time-consuming for researchers and teachers to review student work. For many pages, reading the coded actions yields insight not apparent from the final page artifact. The codes associated with page objects match those created by the human coders. The codes associated with ink strokes could be identified as contributing to corresponding human codes. The analysis of each page takes just a few seconds to run and is done at the click of a button. Importantly, this is the first time that a framework for generating human-understandable summaries of pages has been used in CLP. The framework is written in a modular fashion so that if the human codes or desired output should change, the history action classes can be easily modified or extended.

The analysis program was run on three of 12 pages in 22 students' notebooks, for a total of 66 pages. The pages were chosen as representative examples of the kinds of math problems being used for this math unit. The problems are shown in Table 1. In each problem students were asked to select a multiple choice answer and show how they found their answer.

For all student pages except one, the program generates logically correct action codes. It correctly lists, in order, objects and ink strokes added and removed from the page. Due to limitations in the ink analysis routines, the action codes do not always give the correct summary of a student's interactions. The program, for example, currently can't identify ink that marks multiple choice answers and will try to associate the ink with an object on the page. Identifying ink strokes will be a priority for future work. Results are shown in Table 2, with issues listed that are unrelated to ink identification.

Table 1. Problems used in testing

Page 3	9 times 7 is the same as _____. [Correct answer: 7 x 9]
Page 5	Suzy is 8 years old. Her grandmother is 8 times her age. How old is Suzy's grandmother. [Correct answer: 64]
Page 6	Daniel saves the same amount of money every day. He saves \$56 in a week. How much money does he save each day? [Correct answer: \$8]

Table 2. Summary of results

Artifact Type	Page 3	Page 5	Page 6
Ink only	9 pages, No issues.	7 pages, No issues.	7 pages, No issues.
Array	8 pages. One page throws an error: incorrect array dimensions recorded in history before snapping.	10 pages. After arrays are cut, ink strokes are sometimes incorrectly grouped (associated with different arrays), even though they are part of the same group.	1 page, No issues.
Number Line	4 pages, No issues.	5 pages. After a number line is resized, coded actions do not always show correct size, but this is a minor detail because there was always only 1 number line on page.	14 pages, No issues.
Multiple Types	1 page, No issues.	None	None

6 Future Work

This project successfully began the work to create automatic summaries of each page, and the program is currently usable by researchers and teachers. Comparison of CLP's summaries and human summaries is ongoing, and results will be used to improve CLP's summaries. Some areas for improvement have been identified already.

One feature to implement is a multiple choice page object and history items that would record when such a page object is being marked or erased, instead of just recording ink strokes. With this added functionality, the codes produced for the number line example in Section 3 would match the human codes. With the multiple choice object in place, CLP could give teachers feedback about which answers students chose and whether the answers were correct or not. It also could let teachers know when a student corrected an incorrect answer.

There are many exciting opportunities and options involving handwriting recognition and ink stroke analysis. Since CLP's analysis program can identify ink locations and group ink strokes, the ink

groups could be sent to handwriting recognition programs which might be able to identify higher level actions such as skip counting or equation writing. The skip counting identification work is now being done, and a program has been written that identifies with which row or column a set of ink strokes is associated and uses handwriting recognition to determine the numerical values of the strokes [8]. Incorporating such analysis programs into the history analysis program could yield very informative results. Besides skip counting, some ink patterns we want to identify include equations, incomplete equations, e.g. “ $4 \times 6 =$ ”, and ink that groups objects together, e.g., a circle around student-created images [3,6].

In terms of correctness, there will always be extraneous actions on a page that are unrelated to the student's means of solving the problem. If a student is struggling with the mechanics of using CLP or the tablet, he may move and modify objects by accident. It is also hard to distinguish between ink that has mathematical meaning and decorative or doodling ink. Currently, the only filter employed is ignoring ink strokes that are too small. The CLP notebook is in pen mode by default, so small ink strokes are likely accidental taps on the page. Further filtering will require much more sophisticated artificial intelligence techniques and will be an interesting project for future work.

6 Conclusion

The CLP platform has already proven itself to be a useful tool in the classroom to assist learning and increase student and teacher interaction. This project adds an important component to the tools CLP provides to help teachers gain insight into their students' understanding, namely enabling teachers to view not just the final artifacts students create, but also the process by which the artifacts were created. Researchers also can now use CLP's analysis program as a starting point for analyses instead of needing to hand-code each page—an extremely labor-intensive and time-consuming process. The work described in this document is proof that automatic analysis of each page of student work is

possible, and that even analyses that don't match human analyses completely can still be extremely useful. Extending and improving this analysis is the goal of future work on this topic.

References

- [1] Koile, K., et. Al. Supporting feedback and assessment of digital Ink answers to in-class exercises. In *Proceedings of the Nineteenth Conference on Innovative Applications of AI (IAAI)*, July 24-26, 2007.
- [2] Koile, K., Reider, D., and Rubin, A. INK-12: A pen-based wireless classroom interaction system for K-12. In *The Impact of Tablet PCs and Pen-based Technology on Education: Evidence and Outcomes*. Reed, R. and Berque, D. (editors). Purdue University Press, 2010.
- [3] Koile, K. and Rubin, A. Machine interpretation of students' hand-drawn mathematical representations. In *The Impact of Pen and Touch Technology on Education*. Hammond, T., Valentine, S., Adler, A., and Payton, M. (editors). Springer, 2015.
- [4] Koile, K. and Rubin, A. Tablet-based technology to support students' understanding of division. Presented at the *Workshop on the Impact of Pen and Touch Technology on Education (WIPTTE) 2015*. To appear in published proceedings.
- [5] Koile, K. and Rubin, A. Animated mathematical proofs in elementary education. In *The Impact of Pen and Touch Technology on Education*. Hammond, T., Valentine, S., Adler, A., and Payton, M. (editors). Springer, 2015.
- [6] Maynard, E. D. *Using a Structured Vocabulary to Support Machine Understanding of Student Work*. Master of Engineering Thesis, EECS Department, MIT, Cambridge. February, 2013.
- [7] Rubin, A., Storeygard, J., and Koile, K. Supporting special needs students in drawing mathematical representations. In *The Impact of Pen and Touch Technology on Education*. Hammond, T., Valentine, S., Adler, A., and Payton, M. (editors). Springer, 2015.
- [8] Chandoury, Z. Machine identification of skip counting in Classroom Learning Partner. Undergraduate research opportunity (UROP) project, Spring, 2015.