An Interactive Learning Tool for Early Algebra Education: Design, Implementation and Evaluation

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Siva Meenakshi Renganathan, B. E

Graduate Program in Department of Computer Science

The Ohio State University

2017

Dissertation Committee:

Dr. Christopher Stewart, Advisor

Dr. Arnulfo Perez

Dr. Radu Teoderescu

Abstract

Interactive learning tools allow students to explore STEM concepts deeply, improving educational outcomes. Interactive tools that use cloud computing resources can (1) explore computationally intensive concepts and (2) seamlessly link concepts to graphical representations. When these goals conflict, good design, and implementation principles are needed to (1) preserve interactive response times and (2) uphold curriculum goals. For this paper, we present an interactive cloud-based learning tool that allows students to *interactively* explore algebraic formulas, their corresponding graphs (including axes) and generating data. Our tool prioritizes the following design principles: (1) co-design curriculum and interactive systems support, (2) efficiently collect and share student activities between students, teachers and researchers, and (3) integrate with existing classroom management platforms. Our implementation employs a careful division of work between client-side browsers and cloud servers to provide response times that are within human perception limits. We achieve these design goals by (1) client-side programming for interactive components, (2) sending student activity data to servers at different rates for different sharing types, and (3) align student's data with moodle's (a popular open source classroom management system) schema. Twenty Math teachers were trained to teach Algebra to students using our interactive service. Over eighty percent of teachers trained with our system adopted it. We provide students response times (1) with interactive graphing

less than 15 ms (2) data transfer times of 95% below 0.6 seconds. More broadly, our system has received positive feedback as we have scaled it to over 5 schools, 15 classrooms, and 500 students.

Acknowledgments

I would like to thank all my collaborators, without whom this work would not have been possible.

- Dr. Christopher Stewart
- Dr. Arnulfo Perez
- Bailey
- Tony
- Siva Meenakshi Renganathan.

Table of Contents

Page

List of Tables

Table **Page**

List of Figures

Chapter 1: Introduction

Traditional learning systems divorce STEM concepts from their applications in the real world. They teach STEM concepts and applications separately. Eg: Linear Algebra in Mathematics is not taught alongside Ohm's law in Science which is an application of Algebra. This hinders the students' ability to understand the importance and application of STEM leading to poor student outcomes. Seymour and Hewitt's surveyed students who switched from STEM majors to non-STEM majors in college. Their results indicate that insufficient high school preparation is one of the major reasons students switch from STEM to non-STEM majors [7]. This finding is backed by Lichtenstein's findings that students leave STEM majors due to poor teaching in their pre-engineering courses [4].In addition to STEM, an ACM report states that by 2020, one of every two jobs in the STEM field will be computing [2]. This aligns with the recent inclusion of "Computational Thinking" as a core scientific practice by Next Generation Science Standards [9]. It is important to situate learning effectively through the use of computers and technology and develop technological tools that provide teachers and students the opportunity to enhance their teaching and learning experience. Thus there is an urge for an enhanced K-12 curriculum that explores STEM and Computational Thinking in a tightly coupled and interactive manner.

We proposed an Engineering driven Algebra curriculum for middle and high schoolers to facilitate a better understanding of Algebra and highlights its importance in STEM [6]. This curriculum stimulates learning Algebra by conducting engineering experiments and graphing the data collected. Further, an interactive graphical representation of this data is used to illustrate what data means in the Algebraic and Geometric worlds. We then materialized the curriculum into Algebra lessons and also developed a system to support the interactive graphical functions demanded by our curriculum. For eg, Our first chapter is a Linear Algebra lesson that teaches Linear Algebra through a scientific experiment Ohm's law. Ohm's law is an application of Linear Algebra where Voltage is the independent (x), Current is the dependent (y) and Slope is a function of Resistance. Our curriculum guides students to set up Ohm's apparatus and measure current and voltage as x and y coordinates. This data will then be entered into our system which can graph the data and display the corresponding linear equation ($y = m * x <=> I = \frac{1}{R} * V$). Students should be able to interactively manipulate the graph to explore the effect of slope and intercept on a line.

The implementation of this curriculum demands an interactive smart classroom environment. Slow update times between the input data and graph representation harm educational outcomes. Instant feedback is required for better understanding of the duality of data. Current graphing tools that generate graphs from data input often split up the two and hence are not interactive. They do not guarantee minimal response times as they are not built with interactivity in mind.

We present an interactive learning tool that is developed for a tightly coupled STEM curriculum that allows students to explore STEM concepts alongside its applications. Our tool is a web application that can be accessed by any web browser. With the adoption of Chromebooks and iPads in schools across the nation, students have access to browsers and Internet. They can use our tool to learn STEM concepts in schools and also at home using any computer. Web-based tool also facilitates faster content delivery and flexibility to include more STEM concepts in the future.

Our tool achieves interactivity by efficiently distributing workload between client machines and servers in the cloud. Our JavaScript based graphing component provides interactive response times for all graphing functions. This is achieved by computing and updating interactive graph components in the client without sending data over the network to servers. The curriculum also encourages collaboration. So our system supports the creation of groups amongst students and collaboration and sharing of their work.Teachers can monitor data of all students in real time. These data transfers happen asynchronously without interrupting the interactivity. We also track user interactions with our system like mouse movements and keystrokes for future analysis. Our tool is built on top of moodle - a popular open source course management system. All data collected from users is stored in our database which sits atop moodle's databases. This type of data alignment with classroom management systems enable easy integration and adoption of our tool in schools.

We piloted our curriculum with a group of twenty teachers and trained them to teach Linear Algebra using this curriculum and our tool. Over eighty percent of

trained teachers adopted our system to teach Algebra to their students and now our system is being actively used by over 700 students. We provide interactive response times (1) less than 15 ms for graphing component and (2) tail latency for 95% server requests below 0.6 seconds.

Chapter 2: Curriculum Design

A linear equation in algebra is of the form $y = m * x + c$, where m is the slope and c is the y-intercept of the line. This linear equation can also be represented as a line in a graph by plotting points obtained by giving different values for x and calculating the corresponding values of y. On the other hand given a set of collinear points, the equation of their line can be formed using Two-Point form. Thus, a linear equation can be represented in three paradigms, 1) as a linear equation, 2) as a line in a graph and 3) as (x,y) coordinates of a set of collinear points. An increase or decrease in the slope of the line, rotates the line in anti-clockwise and clockwise directions respectively in the graph. Similarly changes in the y-intercept, shifts the line in an up-down manner in the graph. It is important that students understand the connections between equation, graph and coordinate point paradigms for effective STEM understanding. Our system presents data points, graphs and equations in the same viewport and also allows users to interact with the graph, observe the effect of changing slope and y-intercept in the graph. Our curriculum teaches Linear Algebra exploring two of its application in STEM alongside namely, Ohm's law and a Velocity measurement experiment.

2.0.1 Ohm's Law Experiment

Ohm's law is an important application of Linear Algebra. Ohm's law states that the current through a conductor between two points is directly proportional to the voltage across the two points. Introducing a constant of proportionality (resistance), Ohm's law can be represented as $I = \frac{1}{R} * V$. In Algebra, this equation can be modeled as $y = m * x + 0$, where y is the Current, $m = \frac{1}{R}$ is the slope, x is the Voltage and the y-intercept (c) is 0. Initially with directions from the teacher, students set up Ohm's circuit. Ohm's circuit is a closed circuit formed on a bread board consisting of a voltmeter to measure the voltage, an ammeter to measure the current. In the first phase of this experiment students use a known resistor and different batteries (different voltages) to measure corresponding values of current in the circuit. Each battery and the corresponding current forms an x-y coordinate point. Students enter these x-y coordinate points into our system which presents them an interactive linear graph and displays the linear equation below the graph canvas. Once students gain familiarity with Ohm's law and our system, Phase-2 poses them a small Algebraic quiz. Students are given an unknown resistor and using different batteries like before they will measure corresponding current and find the slope and determine the resistance (Inverse of slope is the resistance).

2.0.2 Velocity measurement Experiment

Placeholder: Ohm's law is an excellent application of Linear Algebra. Ohm's law states that the current through a conductor between two points is directly proportional to the voltage across the two points. Introducing the constant of proportionality, the resistance, Ohm's law can be represented as $I = \frac{1}{R} * V$. In Algebra, this

equation can be modeled as $y = m * x + 0$, where y is the Current, $m = \frac{1}{R}$ is the slope, x is the Voltage and the y-intercept is 0. Initially with directions from the teacher, students set up Ohm's circuit which includes a voltmeter to measure the voltage, an ammeter to measure the current, and a closed circuit formed on a bread board. In the first phase of this experiment students use a known resistor and different batteries to measure corresponding values of current in the circuit. Each battery and the corresponding current forms an x-y coordinate point. Students enter these x-y coordinate points into our system which presents them an interactive linear graph and displays the linear equation below the graph canvas. Once students gain familiarity with Ohm's law and our system, Phase-2 poses them a small Algebraic quiz. Students are given an unknown resistor and using different batteries like before they will measure corresponding current and plot them to find the slope.

Chapter 3: System Design

We developed an interactive system that supports data input and the interactive graphical functions discussed in our curriculum design. Our system is a web application supported by a cloud server that can be accessed from a web browser on any device. Students can sign-up online to use our system. During sign-up process, students will select school, teacher, and grade and based on these mandatory selections they are enrolled in corresponding sections. Once students sign-up, they can log in with their chosen username and password. After logging in, they can access any course materials that their teachers have published or use the link provided to access graphing portal. The graphing portal is divided into four sections: the top left is the 'My Tables' section, top right is the 'Shared Tables' section, bottom left is the 'Data Input' section and bottom right is the 'Graph and Equation' section. My Tables section allows students to create new tables, rename existing tables or delete tables. There are also options to share/unshare tables. Sharing a table exposes it to other students in the same class. Since all these data are stored in the cloud, students can later log in from anywhere from any device and access stored tables. The Shared Tables section displays all tables shared by students in the same. These tables can be imported into My Tables. The Data Input part allows students to type in their experimental data and save them as new tables. This section is also used to

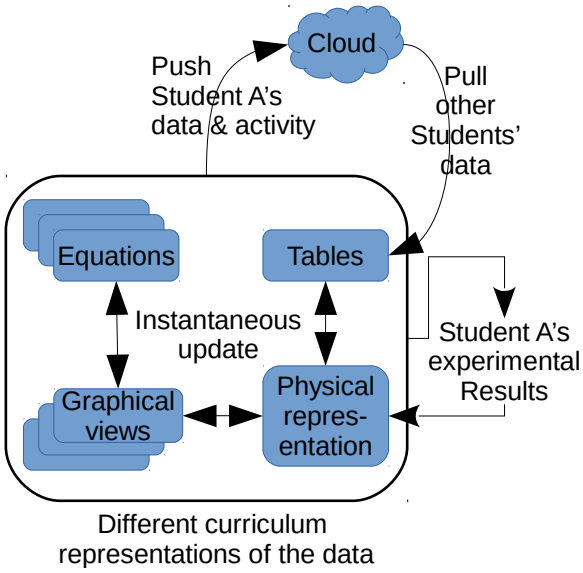

Fig. 3.1: System overview

view/update the data of existing tables under My Tables section. This tabular data is the Physical representation of scientific experiments. Multiple tables in My Tables section can be selected simultaneously and can be combined together. This tabular data can be graphed onto the Graph part using the Graph button. The Graph section has a graph canvas with labeled axes and horizontal and vertical guidelines. Data points from physical representation are plotted on the graph, joined by the line of linear regression. When multiple tables are selected, all points and lines appear in the same graph in different colors one for each source table. The axes rescale automatically based on the input points but can also be updated manually using the update axes button. The Graph section also displays the equations of regression lines and provide a set of sliders for interactive manipulation of data. There are two sliders for every table selected, a slope slider and an intercept slider that can be changed to update the graph and equations instantaneously. Our system also saves the states of

Curriculum Requirements	Number of clicks in	
	Excel	Our system
Input n points	2n	2n
Insert 1 point in between	3	1
Delete 1 point	3	1
Graph points to a line	6	1
View equation of line	3	0
Change slope	-	1
Change intercept	-	1
Update point and update graph	0	0
Add a new line to existing graph	12	1
Delete an existing line from graph	3	1
Update 1 extreme axis	4	2
Update all 4 axis	9	6

Fig. 3.2: Number of interactions required in Excel vs Our system

students work, so they can resume learning without starting over when they log in again.

3.0.1 Curriculum and System Co - Design

The interactive graphical functions demanded by the curriculum can be achieved in minimal interactions at minimal response times only in a co-designed system. Fig.2 shows a comparison of the number of user interactions in Microsoft Excel and our

system for each curriculum requirement. To input n points each having an x and y co-ordinate it takes 2n interactions, (tab press or clicks). An important curriculum requirement is to graph input points and generate a line of regression that fits the input points well. To generate this graph in Excel, more than 5 interactions are required: select data points, insert graph and finally add linear regression trend line to the graph. Also, Excel does not display the equation of trend lines by default and more interactions are needed to view the equation of these lines. In our co-designed system, graphing the points, drawing regressed line and generating the equation of a line, all these steps can be achieved in a single click. Another core curriculum requirement is to be able to manipulate the graph by changing the slope and intercept of lines. Both Excel and our system updates the graph if any of the input points are updated but Excel does not provide users with the option to manipulate graph. Our system allows students to manipulate graphs by changing the slope and intercept using a set of sliders. Updating these sliders simultaneously updates the equation of the line and the data points from which the line was generated. Another curriculum requirement is the ability to simultaneously graph different tables to visually observe the effect of different slopes. After a graph is created in Excel, add data series option can be used to add another set of points to the same graph. While it takes more than ten interactions in Excel to add a new data series, our system can achieve it in a single click. Similarly deleting a data series from the graph is just a click while it takes three interactions in Excel. Finally, the curriculum also requires updating the axes which can be done in our system using two-thirds the number of interactions required in Excel for the same. This illustrates that only a system co-designed with its curriculum can accommodate the curriculum requirements using minimum interactions.

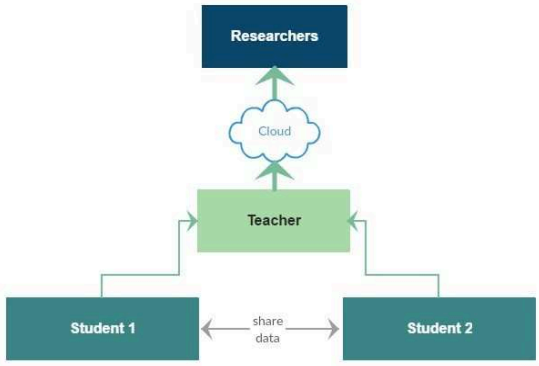

Fig. 3.3: Data flow required by the curriculum

3.0.2 Data transfer between Client and Server

Our Curriculum requires students to create groups among themselves and to share their experimental data within the group. This helps to eliminate the effect of outliers if any, in students experimental data and also promotes the spirit of teamwork in them. Our curriculum demands that teachers be able to view students experimental data to supervise and help students who need assistance. Also, all these experimental data needs to be stored elsewhere than the client machine (in the cloud) so that they can be retrieved later by teachers and students. This data also helps researchers with the research after anonymization. This flow of data demanded by our curriculum is explained in Fig.3.

All data transfer network requests between the client and server are ajax requests (Asynchronous Javascript and XML) except the initial page loads. Ajax requests enable our system to seamlessly exchange data, without clients having to reload an entire page. All our features discussed before like creating tables, retrieving tables,

accessing other students tables, graphing or updating graph can be done from the same page without having to refresh the page. Ajax programming and D3 visualization helped us to create a single page application reducing redundant page loads on the client. Our system also collects students' mouse movements and keystrokes for analyzing and understanding students learning behavior. Sending a network request for every mouse/keyboard interaction increases the number of network requests by a huge factor. At the same time sending all user interactions during logout is not desirable as the number of interactions can grow by orders of magnitude. We batch user interactions with the web page at the client and send them via ajax requests to the server every 140 seconds. The interval 140 seconds was empirically determined optimum from a series of experiments emulating a real classroom traffic. Finally, data sharing between students happen on-demand, (ie) whenever a student wants to access others data, he clicks a local refresh button in Shared Tables section. A click on this refresh button asynchronously fetches all the tables shared by other students in his group. Upon selecting one of the shared tables, the tabular data is pulled from the server. Our results show that the frequency of accessing shared tables is much less than working with one's data. Thus our design reduces unnecessary network communication between server and client to update shared tables periodically.

3.0.3 Integration with Classroom Management Systems

Moodle and Blackboard are very popular classroom management systems of which moodle is open sourced. Our tool is built alongside moodle using many of the course

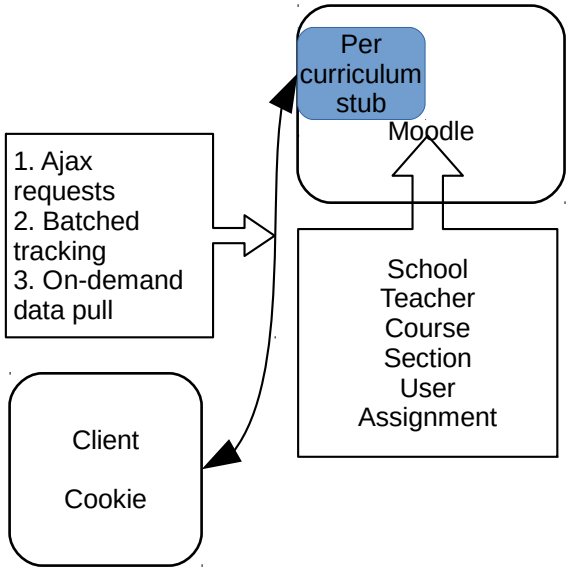

Fig. 3.4: Data transfer between client and server

management features that moodle provides. Typically classroom management systems handle students in one educational institution but we extended moodle to manage classes across multiple educational institutions (schools). We introduced another layer of abstraction on top of moodle's database which distinguishes students from different schools using a unique school-id identifier. We use moodle's secure login features for students to log in and we have tweaked moodle's course enrollment api to enroll students from different into our system. Our system supports many schools, with many teachers in each school and each teacher can handle many grades/sections in the same school. We use an attribute called class-id in addition to moodle's databases to achieve this one to many relationships. With this unique class-id, a teacher in one school can handle different sections in the same grade or different grades in same or different schools. Our hierarchy is described in Fig 4. We also have additional databases to store students' experimental data, meta-data about students' tables and

students' usage patterns (mouse/keystrokes).

When a student logs in, the server sets a cookie in the client's web browser that contains values like class-id and user-id. These ids are used by the translation stub in the server to identify future requests from this student. Any tables that he/she creates is tied to this unique class-id and user-id and stored in the experiment-data database. This data is accessible only through the user-id or the class-id. A table can be exposed to others with the same class-id by setting a shared flag. Using class-id and shared flag we provide data flow among students in the same class. Upon login, teachers will be presented with a page to choose the class she is handling currently. Teacher's selection on this page updates the class-id value in the cookie. A teacher by default has access to all student's data in each class and this is done by overriding the shared flag in the database.

Our curriculum contains multiple scientific experiments and each experiment can be regarded as a chapter. The graphing and data input elements for each chapter vary only with respect to axes labels and units (voltage, current and time, distance). Hence these chapters are presented as tabs in a single web page. All data transfers between the client and server also have a chapter-id parameter in addition to class-id and user-id. Students' data are grouped by chapter ids in the database. When accessing a chapter, data-tables related to only that chapter are shown as to not overwhelm students with a large number of tables.

Chapter 4: Deployment and Analysis

We piloted our system in summer 2016 with 20 teachers. During this pilot session, we trained the teachers to practice our curriculum and use our system effectively. The teachers were given an algebra quiz before and after their training with our curriculum. Quiz results show that training with our system can increase their understanding of Algebra and introduce them to Science and Engineering. We updated our curriculum with feedback from teachers after the pilot session. This updated system is deployed live and more than 80 percent of the teachers that piloted our curriculum adopted it. They use our system now to teach Algebra to their students. Currently, our system is being used by more than 700 students from five schools in the Columbus area.

The weblogs from students usage were analyzed to understand students' usage pattern and to look for potential optimizations in our system. Weblogs analysis of different classroom sessions shows that they follow the same pattern. A typical classroom session in the school is about 45 minutes long and the student activity across a classroom session in terms of the number of network requests per min is shown in Fig.5. Initially, it takes few minutes for the students to set up their computers and to navigate to our portal's login page. This corresponds to the initial requests in the figure between 0 to 5 minutes. Then the students type in their usernames and

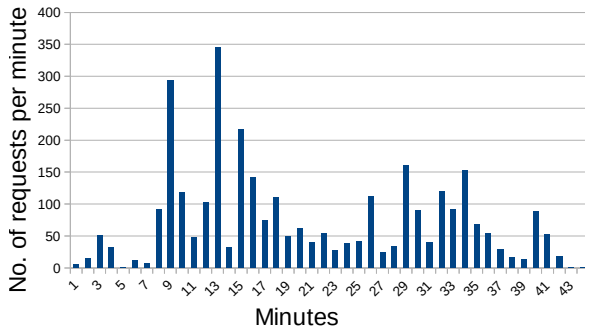

Fig. 4.1: No. of requests received per minute by server in a classroom session

passwords and log-in. These login requests correspond to the spike in server traffic between 7 to 11 minutes. Since more content is served post login, the server is busier than it was while serving login pages. Once students are logged in they enter their experimental data and save tables. Since creating, updating and sharing tables are all handled as separate ajax requests by the client to the server, we notice a period of peak network requests during this period. There are spikes at regular interval that corresponds to the user-tracking data sent by the client.

Chapter 5: Evaluation and Optimizations

We evaluated our interactive graphical client for response times and determined the variation in response times with increase in number of points in the graph. Fig.6 shows the variation in response times for every slope/intercept update on the graph for different number of points, the client was a Chromebook provided in the school. When the graph button is clicked, the JavaScript client does a linear regression on the points in the graph and draws the regression line. Also upon every update to the graph, the new slope and intercept values from the slider are used to calculate the updated (x,y) coordinates for points and draw the new line. All updates to the graph are handled by the client without any interaction with the server. We provide response times of few milliseconds for every graph interactions without any server interaction by using the popular JavaScript visualization library D3.

From our analysis, we determined that our server is busy most of time during a classroom session and so we worked on ways to optimize our server. Following are some of the optimizations implemented in our apache based server to reduce query response times. 1.Caching, 2.Compression and 3.Images and CSS optimizations. Apache caching module (mod_cache) instructs the browser to cache pages so that they can be served from the cache in the future without having to send a request to the server. As discussed earlier our web application is a single page application

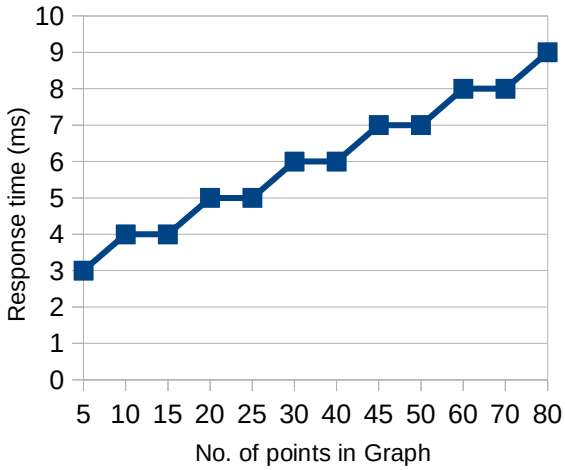

Fig. 5.1: Interactive web client tightly coupled to scientific experiments

with asynchronous data requests. This facilitates caching since all the static content can be cached at the client and only data requests can be served by the server. These data requests couldn't be cached anyway because there are initiated by user interactions. . Our cached client on an average serves more than 90% of static page elements and thereby reducing the load on server. For example, 27 of 29 page elements in our login page is served by browser cache. Finally we distribute this random usernames and passwords to students which they can use to login directly. This approach helps us achieve complete anonymity and also better response times as there is no cipher/encryption involved. Apache compression module (mod_deflate) uses gzip compression to reduce the number of bytes transferred over network. Reducing the number of bytes sent over the network can reduce the round-trip-time for every data request. Moodle provides the admin an option to customize themes and background for all pages. Through experiments we found out that themes with background images and more CSS styling take significantly longer times to reach the

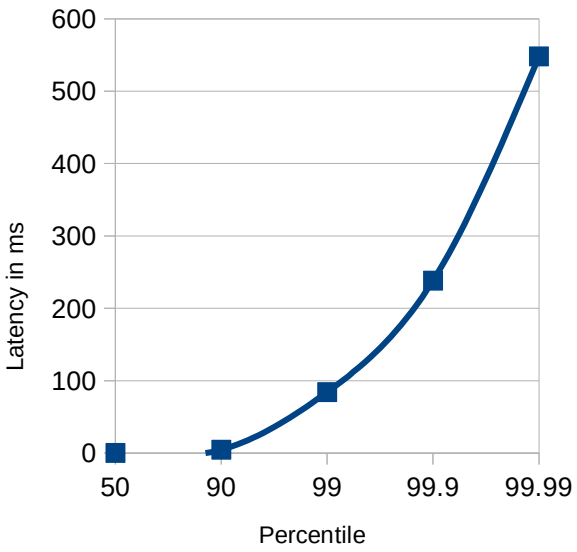

Fig. 5.2: Interactive web client tightly coupled to scientific experiments

client and for the client to apply those stylings. We have currently disabled themes to provide a very simplistic but functional front-end. This simple UI reduces page size and page loading overhead by over 30%.

Our tail latency distribution after these optimizations is shown in Fig.7. Our server serves more than 90% of the requests in less than 0.1 seconds.

Chapter 6: Related Work

The advantages of web based learning is that it not only provides visual interactivity with the resources that encourages students to actively participate in the learning process but also facilitates accessing information about the sophisticated interactive resources [8].

Approximate computing:
In software-driven approximation bound by quality limits, speeding up the processing of data and reducing delay has been explored in several works such as:

- ApproxHadoop [1]: In this approach, the processing delay is reduced by dropping map tasks to speed up map-reduce computations, lowering the quality of final answers

- Ubora [3]: This work proposes an approach to measure the effect of slow running components on the quality of answers, memorize and improve the computation.

- Sprinting [5]: This approach explores increasing processing rates by exceeding budgets for short bursts before reverting back to safe processing rates.

Chapter 7: Conclusion and future work

We developed an engineering driven scientific curriculum and co-designed a system that supports the interactive graphing functions required by the curriculum. We addressed the problems in designing such a system by our design principles Client programming, asynchronous data-sharing at different intervals and tight integration with classroom management systems for easy adoption. we also looked for potential opportunities to optimize the system and thereby reduced response times. Our curriculum can be extended to teach more Mathematical concepts like Polynomial Algebra and Geometry. Our system can be extended to support these new functions with more JavaScript programming. In general, our system can be used as a prototype for building an interactive classroom application which can support students from many schools simultaneously.

Bibliography

[1] Inigo Goiri, Ricardo Bianchini, Santosh Nagarakatte, and Thu D Nguyen. Approxhadoop: Bringing approximations to mapreduce frameworks. In *ACM SIGARCH Computer Architecture News*, volume 43, pages 383–397. ACM, 2015.

[2] Lisa C. Kaczmarczyk and Renee Dopplick. Acm report: preparing students for computing workforce needs in the u.s. *SIGCSE Bulletin*, 46:8, 2014.

[3] Jaimie Kelley, Christopher Stewart, Nathaniel Morris, Devesh Tiwari, Yuxiong He, and Sameh Elnikety. Measuring and managing answer quality for online data-intensive services. In *Autonomic Computing (ICAC), 2015 IEEE International Conference on*, pages 167–176. IEEE, 2015.

[4] Gary Lichtenstein, H Loshbaugh, B Claar, T Bailey, and S Sheppard. Should i stay or should i go? engineering students persistence is based on little experience or data. In *Proceedings of the American Society for Engineering Education Annual Conference*, pages 24–27, 2007.

[5] Nathaniel Morris, Siva Meenakshi Renganathan, Christopher Stewart, Robert Birke, and Lydia Chen. Sprint ability: How well does your software exploit bursts in processing capacity? In *Autonomic Computing (ICAC), 2016 IEEE International Conference on*, pages 173–178. IEEE, 2016.

[6] Arnulfo Perez, Kathy Malone, Siva Meenakshi Renganathan, and Kimberly Groshong. Computer modeling and programming in algebra. In *Proceedings of the 8th International Conference on Computer Supported Education - Volume 1: CSEDU,*, pages 281–286, 2016.

[7] Elaine Seymour and Nancy M Hewitt. Talking about leaving, 1997.

[8] Judy Sheard, Jason Ceddia, John Hurst, and Juhani Tuovinen. Inferring student learning behaviour from website interactions: A usage analysis. *Education and Information Technologies*, 8(3):245–266, 2003.

[9] David Weintrop, Elham Beheshti, Michael Horn, Kai Orton, Kemi Jona, Laura Trouille, and Uri Wilensky. Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1):127–147, 2016.