ISSN: 2473-3857
v3 n4357

Computational thinking and application – a cross-curricular approach to teach computer science in high school chemistry classrooms

Razieh Fathi
Raziehfa@buffalo.edu
Department of Computer Science and Engineering
University of Buffalo
Buffalo, NY 14260, USA

Daniel Hildreth
dhildreth@buffaloschools.org
South Park High School
Buffalo, NY 14220, USA

Abstract

According to the "CSforALL" movement that works to enable students at the national level to gain computer science literacy during their education, there is a growing realization of the importance of having computing education in all high school classrooms in the United States. To make this important gain, many attempts and techniques are being explored. We provide one possible solution, a technique to teach computer-programming concepts as part of computational thinking skills in a high school chemistry classroom. First, we describe our motive in providing a cross-curricular approach. We discuss ways a cross-curricular approach can help students reach a deeper understanding of both subjects (chemistry and computer science). Second, we describe using classroom backward design as seen in Wiggins and McTighe's *Understanding by Design* methodology. Using this method, we bring computer-programming concepts to a high school chemistry classroom using Microsoft Excel. Finally, we summarize the current status and future plans for further development of the project.

Keywords: high school chemistry, computational thinking, computer programming concepts, cross-curricular activities.

1. INTRODUCTION

Increases in the usage of information technology tools prioritize the importance of a computer literate society. This literacy is especially important when we think about the education of a new generation in that society.

Information technology has had a significant impact in the lives of this generation, especially an economic one.

A strong economy requires computer literate members that can increase the economic value of that society. Having knowledge of computers is so important that Rushkoff in "Program or be programmed" stated in 2010, "in the emerging, highly programmed landscape ahead, you will either create the software or you will be the software."(p.48)

The idea of teaching computer science at all levels began with Perish in 1962 (Guzdial, 2008). Since then, there has been a large movement to teach computer science and

information technology for all levels. The most comprehensive view of this attempt, in the US, came from then President Obama's administration, which launched the "CSForAll" initiative in 2016. This program provided increased support from a variety of places over the following year (Chapman, 2017).

We can see some of the positive results of the computer science education movement from high enrollment rates for computer science at the college level (Zweben & Bizot, 2015). However, there is still a significant lack of exposure in computer science for K-12 students (Lee, Dombrowski, & Angel, 2017).

This lack of exposure is more apparent when we deal with students from low socio-economic backgrounds (Heitin, 2015).

One main reason for this is the shortage of trained high school and middle school computer science teachers in these school districts. This proves to be a significant challenge for having "CSForAll".

The authors of this study believe one micro-scale path for solving this challenge would be to help STEM teachers embed computational thinking techniques within their current STEM curriculum. According to Wing (2006), computational thinking not only belongs to computer scientists, but it belongs to everyone. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability (Caspersen & Nowack, 2013; Wing, 2006).

Bringing computational thinking to the STEM classroom can be challenging and overwhelming for both students and STEM teachers. The best way to deal with this issue is to use an accessible tool that could simplify computational thinking concepts.

In this paper, we describe our preparation of course materials for the chemistry classroom using the spreadsheet program, Microsoft Excel. This software platform can introduce computer-programming concepts efficiently in the science classroom for four reasons: first, the accessible nature of the software and the easy-to-learn nature of that software, second, the general familiarity of STEM teachers with the software, and third, the interactive environment of the spreadsheet itself. Finally, many business companies in the US have been using Excel as an IS tool. Exposure and training with support from this software will give the students an advantage when looking for a job or pursuing higher education. These factors support the use and implementation within the high school environment. For our purposes, we designed the materials to enhance the typical high school chemistry classroom in New York State.

Next, we describe our motivation in detail, the application of backward classroom course design, the teaching and application of the Excel platform, and finally, the current status and plans for the project.

2. Background

We started our effort by identifying common computational similarities between CS and chemistry before developing any content. Throughout this process, our thesis served as a guiding principle for our efforts of refinement and concrete design of the material. We employed backward course design to prepare our content. We believe this design is relevant to the type of result we are looking for.

Having computing in all classrooms and the need to pay close attention to the classroom concepts for deeper learning, lead us to propose a cross-curricular strategy as an important solution for reaching the goal of having "CSforAll". With a cross-curricular approach, not only is there an opportunity to have the assistance of STEM teachers, but we also create an environment for deeper learning. "We can accelerate deeper learning by consolidating teacher efforts and combining relevant contents. In fact, in this way we open new pathways of knowledge" (Johnson, 2013). Finding a way to bring computational thinking concepts into the classroom will open an arena for deeper learning for all students involved.

3. THE CLASSROOM: BACKWARD DESIGN

One of the most familiar approaches in course design is beginning with the most convenient methodology for teaching the content. This means the focus is on teaching methods rather than teaching contents. Many teachers use "A taxonomy for learning, teaching, and assessing" (also known as Bloom's Revised Taxonomy) as a guide to defining the goal (Anderson & Krathwohl, 2001). The following constitute the categories within the revised taxonomy along with a brief description of its meaning:

- Remember: the ability to recover and access knowledge from long-term memory
- Understand: the ability to understand, interpret, classify, summarize and compare, and to construct meaning
- Apply: the ability to perform an unfamiliar task by applying previous knowledge to a new situation or problem
- Analyze: ability to organize, differentiate and attribute
- Evaluate: ability to judge, reason, and critique
- Create: ability to innovate and produce new knowledge

Using the taxonomy to identify the level of students' engagement is a good starting point, but it is not the whole picture. Within this larger picture, and just as important, is the pedagogical design of the lesson. As educators designed classes with the intention of having students achieve the highest level within Bloom's taxonomy, but without a means of advancing students, this design proved meaningless. "Teaching is a means to an end. Having a clear goal helps as educators to focus our planning and guide purposeful action toward the intended results" (Wiggins & McTighe, 2005, p.7). Wiggins and McTighe (2005), in their Backward Design framework, reverse the typical approach in order to focus primarily on the desired learning outcomes, and then based on this, to decide on the deliverance of that material. This design approach has three main steps:

- Define the aimed goals
- Find out the convenient criteria to monitor students' progress
- Aim for the relevant instructional methodologies

The authors of this study have chosen to use this design framework because of its focus on outcomes and Bloom's revised taxonomy: to define the goal and form a plan to improve student reasoning through computational thinking.

3.1. DEFINE THE GOAL

The goal for this project is to improve student achievement. We plan this improvement through the introduction of computational thinking into the curriculum of an urban high school chemistry classroom. Computational thinking is logical and precise. These are very much the same skill sets needed when studying science, including chemistry. Chemistry is very abstract, which adds to the complexity. In some urban schools, this will be the student's first encounter with an abstract science like chemistry. Their previous exposure to science was with biology and earth science; both curriculums are very concrete and relatable. The authors feel computational thinking will give the students a tool to aid them in their understanding and conceptualization of this difficult course.

For the purposes of this study, our focus within the New York State High School chemistry curriculum will be on the topic of stoichiometric. This topic combines aspects of the chemistry curriculum (measurements, dimensional analysis, atomic mass, the mole concept, chemical formulas, equations, etc.) in such a way that students need to be very logical, precise, and methodical. The authors feel this makes this unit an ideal one for the introduction of computational thinking strategies. If students can master this topic using computational thinking skills, they will be more successful with the unit, the summative assessments, the other units within this curriculum and perhaps other classes. Therefore, for the purposes of this study, narrowing the goal from an overarching statement of improving student achievement to one of improving comprehension and retention of the topic of stoichiometry makes more sense and gives reasonable allowances for the narrow goal. If the students can master the stoichiometry unit with computer science skills, then it stands to reason that they would be successful transferring those skills to other units within the curriculum, thereby improving student achievement, our overarching goal.

3.2. ASSESSMENT EVIDENCE

According to the backward design model, once a goal has been defined, assessment criteria needs to be established to monitor student's progress. These assessment criteria need to be convenient and relevant. Because this model can be implemented at any level within the curriculum, from a single unit to the entire year, the assessment pieces will include a pre-test/post-test component for the unit, then a variety of formative and summative assessment pieces addressing the concepts covered, both in chemistry and computer science.

The authors have determined several formative pieces that would contribute to our overall evaluation of the student's progress towards the

main goal. Short-answer quizzes, free-response questions, homework assignments, practice problems, and laboratory write-ups are the other assessment pieces that we will be evaluating (Bowen, 2017). One example of an assessment that addresses both the chemistry and computer science skills and is relevant to the backward design model is the laboratory write-up. Laboratory write-ups appeal to the application of knowledge and concepts in new and complex contexts. The quizzes provide a more traditional testing venue the students must be familiar with in the standardized state exam given at the end of the year. The free-response questions, homework assignments, and practice problems will all provide ongoing evaluation of the students' progress through the course of the project. We feel that all the assessment types are important to monitor the students' progress as we work towards the goal of improved student achievement in chemistry.

Also within this assessment portfolio, the final statewide administered test in chemistry will be used to determine the level of comprehension and retention the students achieve using the computer science skills taught, as compared to previous years in which the CS skills were absent for the stoichiometric unit. Because the statewide final assessment piece is given at the end of the year, this can also inform our understanding of their comprehension through long-term retention. This data will inform the instructional practices of the teacher for the following year, and determine the level of comprehension the students have achieved using the computer science skills. This information will give this teacher insight as to the viability of the computer science skills taught and whether they should be incorporated into other units within the chemistry curriculum.

3.3. RELEVANT INSTRUCTIONAL METHODOLOGIES

In our effort, we will use three strategies that encourage students to actively explore the new concepts and this unfamiliar way of problem solving. Based on the educational design principals, one effective way of doing it is through interactive lecturing, a form of active learning. This method "promote[s] the kind of cognitive work identified as necessary for learning" (Brame, 2016, para.2). Within this methodology, there are numerous strategies that can be employed to facilitate the cognitive construction that is sought through our interactive lectures. The specific strategies employed will be discussed in future iterations of this project.

Another strategy we will employ during this project is the use of discussions. "Engaging students in discussion deepens their learning and motivation by propelling them to develop their own views" (McDaniel, 2017, para.1). Discussion of the principles and the more practical steps to solving problems, offers students insight into each other's thinking processes to augment their own cognitive construction for learning new material. Specific strategies planned and employed will be discussed in later versions of this project.

The third, and final, methodology we will employ in the pursuit of the goal is the cooperative learning strategy. Running parallel with the other two methodologies, this one will support the laboratory portion of the course. Based According to Johnson and Johnson (1989), through the instructional process of cooperative learning, students' work in small group to maximize their own and each other's learning. The laboratory setting is one in which it is very difficult for a single student to accomplish the required procedure and record the necessary data on their own. This facilitates the need for small cooperative groups. Within these small groups, students often realize their strengths and weakness and learn to rely on each other for their cognitive development with new concepts. The shared burden of accomplishing laboratory activity tasks is merely a vehicle to promote a cooperative environment. It is that cooperative environment that promotes an enhanced learning situation for all involved.

We feel these methodologies, when employed together, offer the best solution for teaching computational thinking in a secondary classroom of chemistry students.

4. EDUCATION OF COMPUTER PROGRAMMING LANGUAGES

According to Lee et al., (2017), there have been some professional teaching programs in New Mexico, (NM-CSforAll), that serve STEM teachers with some CS training for a dual credit in computer science course.

We can think about that approach in a macro level, but what if we think about an approach that can serve STEM teachers, especially the ones that work in low- income schools? We should consider their challenges, like

insufficient facilities and working with a specific type of population.

Even with providing some CS training, we cannot have high expectations for STEM teachers to teach programming and algorithms. Even in with CS teachers, based on a work by (Tatsumi, Nakano, Tajitsu, Okumura, & Harada, 2009) most teachers at K-12 level do not consider teaching programming languages or algorithms. These teachers tend to set their goals too high, with expectations too high for the students and using textbooks that are too difficult for beginners.

Considering this reality, we should think about the possible approaches to teach a complete novice programmer how computers work and the building blocks of most programming languages.
In this section, we introduce some common approaches to teach computer programming. Then we continue with describing how we can use spreadsheets as a platform to learn computer programming.

Tatsumi et al. (2009) categorized two methods of teaching programming languages to a novice learner, which are as follows:

- Provide a functioning program and let the students change various parameters or pieces to let them see how those changes affect the outcome. In this approach, one can take a relatively complex code from the beginning and let the students work on particular parts and gain understanding of those particular details of the programming language in question.

- Explain how parts of an algorithm should work in flow charts or some other manner and explain how that should be translated in a specific language. Then ask students to write a piece of working code from scratch.

The second method is considered the most effective approach for a novice programmer, but it is also challenging. Students may be puzzled and misled by several minor details. They need to pay very close attention to the smallest details of a program such as commas, periods and other delimiters.

In most scenarios, the troubles that come from these details can be cumbersome and discourage novice learners from continuing to learn any programming.

One quick and effective solution to teach the same concepts is to use software that can reduce the complexity of working with details but still be able to teach the same concepts in a most interactive way.

This is why we argue using a spreadsheet can be of great assistance for both teachers and K-12 students.

4.1. EXCEL AS A TOOL

As previously discussed, students can learn the concepts of computer science and programming more quickly and with least difficulty if they are given a baseline in problem-solving skills and related concepts that avoid using syntactical details of an implementation solution. Through using excel, the discovery of several computer-programming concepts becomes intuitive and students can realize the functionality and existence through the elements in spreadsheets.

Danielle Bernstein (1990) pointed out that, one can use software packages to teach such concepts as files, records, structures, Booleans, memory, and data types "These topics can be examined without the overhead of extensive program planning or syntax problems that get in the way of a beginner" (Frenkel, 1990, p. 34).

In addition to Bernstein (1990), more concepts, such as variables, operators and order of precedence, functions and parameters, pointers, control structures, and designing for change, are featured in spreadsheets making them a good platform to teach these concepts.

Since a spreadsheet cell can contain an alphanumeric label, a numeric value or a formula the concept of data types flows naturally (Kolesar & Allen, 1995).

By setting up different format for displaying a number, we can introduce easily the data types such as integer, floating point, double.

Without encountering syntactic difficulties, teaching the concept of formatting with spreadsheets can become very intuitive for students.

Introducing different formulas through built-in functions in excel, opens up a productive workspace to discuss the concepts related to

function, parameters and conditionals as the main building blocks of programming.

As we planned our course content we began, from the computer science standpoint, by introducing the variable and constant. Then we introduced the concepts of operators, expressions and order of precedents, then Boolean expressions with actual exercises stemming from actual content for the high school chemistry classroom. Conditionals and functions are two other concepts that we designed to teach with spreadsheets.

Teaching the concept of the loop using a spreadsheet can be challenging, but considering the fact of usage of copying formulae with relative cell references prepare us with a good tool to demonstrate the concept (Kolesar & Allen, 1995).

We plan to use flowcharts as a tool to teach algorithms and as a way to guide students to use this as a complementary technique to familiarize themselves with the actual world of programming and computer science.

In our effort based on a high school chemistry curriculum, we were able to match all the main concepts that are discussed in CS1 programming classrooms at the high school level.

We developed four main final projects from the same actual contents in high school chemistry curriculum, in continuation with this preparation.

4.2. SPREADSHEETS AND CAREER ENHANCEMENT

Spreadsheets have been used for an extended variety of purposes, from inventory administration to educational applications and from scientific modeling to financial systems.

Spreadsheets are one of the tools that are widely used in the work place. According to Winston in 2001 90% of all analysts in industry perform calculations in spreadsheets.

In relation to the same idea Panko in 2006 estimated that 95% of U.S. firms, and 80% in Europe, use spreadsheets in some form for financial reporting. Business analysts using spreadsheets usually have very limited (if any) training as a programmer.

On the other hand, Ko et al. (2011) brought up the notion that spreadsheet users are end-user programmers. With this in mind, these groups are becoming involved with problems such as "identifying faults, debugging, or understanding someone else's code." (Hermans, Pinzger, & Van Deursen, 2011, p.451).

All of these evidences shows the importance of learning Excel for a new generation and the contribution of excel to the IS society.

The win-win prospect of using Excel is that while the program provides an effective gateway into the world of computer programming, it also serves students' broader needs by providing skills that can be successfully applied to any number of career paths.

5. CONCLUSIONS AND FUTURE PLANS

In this paper, we offer an approach to teaching computer programming in a practical and effective way that is geared towards the needs of local urban public schools.

We have focused on Excel as a tool to teach programming concepts. In parallel to learning computer-programming concepts, learning Excel can be a pathway for career enhancement for the targeted students.

We have described the role that knowledge of Excel plays in the industries and businesses.

We embed these important learning experiences within a science classroom, where we can solve part of the problem for having "CSforAll" in all schools.

For future enhancement of our approach, we plan to implement the CS content materials in a High School chemistry classroom this coming fall semester

We believe our cross-curricular project offers important paths for future exploration. In particular, we wish to investigate the following research questions:

- How can this approach be integrated into other science subjects?
- How can we develop effective and efficient teacher training for this approach?

6. ACKNOWLEDGEMENTS

The authors would like to thank Jim Teresco for his invaluable feedback.

ISSN: 2473-3857
v3 n4357

7. REFERENCES

Anderson, L. W., & Krathwohl, D. R. (2001). A taxonomy from learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Longman, New York.

Bernstein, R. D. (1990). Understanding Spreadsheets: Effects of Power Users' Learning Style and Computer Training on Performance and Mental Model Acquisition. Doctoral Dissertation. Rutgers University, New Brunswick, NJ, USA.

Bowen, R. S. (2017). Understanding by Design. Vanderbilt University Center for Teaching. Retrieved July 15, 2017 from https://cft.vanderbilt.edu/understanding-by-design/

Brame, C. (2016). Active learning. Vanderbilt University Center for Teaching. Retrieved July 15, 2017 from https://cft.vanderbilt.edu/active-learning/

Caspersen, M. E., & Nowack, P. (2013, January). Computational thinking and practice: A generic approach to computing in Danish high schools. In *Proceedings of the Fifteenth Australasian Computing Education Conference-Volume 136* (pp. 137-143). Australian Computer Society, Inc.

Chapman, G. (2017, March). Inspire, Innovate, Improve!: What does this mean for CS for All?. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 1-1). ACM, New York, NY, USA.

Frenkel, K.A. (1990). Women and Computing. Communications of the ACM, 23(11), 34-46.

Guzdial, M. (2008). Education Paving the way for computational thinking. *Communications of the ACM*, *51*(8), 25-27.

Heitin, L. (2015, August 26). Computer Science; "Searching for Computer Science: Access and Barriers in U.S. K-12 Education". *Education Week*, *35*(2), 5.

Hermans, F., Pinzger, M., & Van Deursen, A. (2011, May). Supporting professional spreadsheet users by generating leveled dataflow diagrams. In *Proceedings of the 33rd International Conference on Software Engineering* (pp. 451-460). ACM, New York, NY, USA.

Johnson, B. (2013, January 15). Deeper Learning: Why Cross-Curricular Teaching is Essential. Retrieved July 15, 2017 from https://www.edutopia.org/blog/cross-curricular-teaching-deeper-learning-ben-johnson

Johnson, D & Johnson, R. (1989). Cooperation and competition: Theory and research. Interaction Book Company, Edina.

Ko, A. J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., Scaffidi, C., Lawrance, J., Lieberman, H., Myers, B., & Rosson, M. B. (2011). The state of the art in end-user software engineering. *ACM Computing Surveys (CSUR)*, *43*(3), 21.

Kolesar, M. V., & Allan, V. H. (1995, March). Teaching computer science concepts and problem solving with a spreadsheet. In *ACM SIGCSE Bulletin 27*(1) (pp. 10-13). ACM, New York, NY, USA.

Lee, I. A., Psaila Dombrowski, M., & Angel, E. (2017, March). Preparing STEM Teachers to offer New Mexico Computer Science for All. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 363-368). ACM, New York, NY, USA.

McDaniel, R. (2017). Discussions. Retrieved July 15,2017 from https://cft.vanderbilt.edu//cft/guides-sub-pages/discussions/

Panko, R. (2006). Facing the problem of spreadsheet errors. *Decision Line*, *37*(5): 8-10.

Perlis, A. (1962): The computer in the university. In Computers and the World of the Future, 180-219. Greenberger, M. (ed.). MIT Press.

Rushkoff, D. (2010). Program or Be Programmed – Ten Commands for a Digital Age. OR Books, New York.

Tatsumi, T., Nakano, Y., Tajitsu, K., Okumura, H., & Harada, Y. (2009, November). Incorporating music into the study of algorithms and computer programming.

ISSN: 2473-3857
v3 n4357

In *Proceedings of the 2nd Workshop on Child, Computer and Interaction* (p. 19). ACM, New York, NY, USA.

Wiggins, G., & McTighe, J. (2005). Understanding by design (2nd ed.). alexandria, VA: Association for supervision and curriculum development ASCD. *Colombian Applied Linguistics Journal*, *19*(1): 140-142.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Winston, W.L (2001). Executive education opportunities. *OR/MS Today, 28*(4): 8-10.

Zweben, S., & Bizot, B. (2015). Taulbee Survey: Continued Booming Undergraduate CS Enrollment; Doctoral Degree Production Dips Slightly, Computing Research News, 28(5). Retrieved from http://cra.org/wp-content/uploads/2016/05/2015-Taulbee Survey.pdf

ISSN: 2473-3857
v3 n4357